SOME CONTRIBUTIONS TO LATIN HYPERCUBE DESIGNS, IRREGULAR REGION SMOOTHING AND UNCERTAINTY QUANTIFICATION

A Thesis
Presented to
The Academic Faculty

by

Huizhi Xie

In Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy in the
H. Milton Stewart School of Industrial and Systems Engineering

Georgia Institute of Technology
August 2012

SOME CONTRIBUTIONS TO LATIN HYPERCUBE DESIGNS, IRREGULAR REGION SMOOTHING AND UNCERTAINTY QUANTIFICATION

Approved by:

Professor C. F. Jeff Wu, Advisor
H. Milton Stewart School of Industrial
and Systems Engineering
Georgia Institute of Technology

Professor Yajun Mei
H. Milton Stewart School of Industrial
and Systems Engineering
Georgia Institute of Technology

Professor Xiaoming Huo
H. Milton Stewart School of Industrial
and Systems Engineering
Georgia Institute of Technology

Professor Godfried Augenbroe
College of Architecture
Georgia Institute of Technology

Professor Roshan Joseph Vengazhiyil
H. Milton Stewart School of Industrial
and Systems Engineering
Georgia Institute of Technology

Date Approved: 20 April 2012

To my parents.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

ACKNOWLEDGEMENTS

Deep appreciation is sincerely given to whoever has educated, supported and inspired me. I would not have been able to successfully complete my PhD study without them.

First and foremost, I would like to express my deepest gratitude to my advisor Professor C. F. Jeff Wu. He has guided, supported and encouraged me throughout my PhD study with incredible patience. I am not only inspired by his passion and deep insight on academic research, but also influenced by his professional ethics, aspiring life attitudes and broad knowledge. All these will be invaluable treasure in my career.

I am extremely grateful to Professor Xiaoming Huo for supporting my academic research. His enthusiasm, deep insight into statistics, and originality have made it very pleasing to work with him. His guidance has dramatically enhanced my theoretical understanding of statistics.

I would also like to thank Professor Roshan Joesph Vengazhiyil for support and help during my PhD study. His incredible originality has made research such an attractive part of life. I wish to thank Professor Godfried Augenbroe for serving on my thesis committee and leading the EFRISEED project, which makes my humble contribution of applying statistics to quantify the uncertainties in building energy simulations possible. My thanks also go to Professor Yajun Mei for serving on my thesis committee and providing many insightful suggestions.

I wish to thank Professor Peter Z. G. Qian for the support of my academic research. I have benefited a lot from him in the research of design for computer experiments.

I am grateful to Dr. Yasuo Amemiya for supporting my internship at I.B.M. T. J. Watson Research Center and to Dr. Rajesh Jugulum for supporting my internship at Citigroup.

Many thanks go to my lab members Dr. Tirthankar Dasgupta, Dr. Ying Hung, Dr.

Xinwei Deng, Dr. Lulu Kang, Dr. Nagesh Adiga, Matthias H. Y. Tan, Heng Su and Rui Tuo. It is a great honor to get to know these talented people and the friendship with them is deeply valued.

My other personal friends at Georgia Tech are also an important part of my PhD life. They include but are not limited to Dr. Huijing Jiang, Yibiao Lv, Carlos Valencia, Dr. Lingyang Ruan, Hao Wu, Calvin Chen Xu, and Fei Zhao. The association with them, both academically and casually, has significantly enriched my PhD journey. I am especially grateful to Zhuzhou Fu. I consider myself extremely lucky to get to know such a considerate girl, who is definitely a prominent highlight of my life at Georgia Tech.

At last but not least, I would like to express my sincere gratitude to my parents for their unconditional love and support. I inherit from them the moral of being up-and-coming, persistent, and optimistic. I would like to summarize my endless word to them using one famous Chinese saying: “Nothing is as touching as parents’ love”. This thesis is dedicated to them.

SUMMARY

This thesis consists of three parts. In part I, we propose a general framework for slicing Latin hypercube designs in computer experiments and focus on a special case called doubly sliced Latin hypercube design (DSLHD). In part II, a completely-data-driven smoothing technique is proposed for irregular region smoothing and its properties are investigated. Part III deals with uncertainty quantification of energy assessment in the field of building technology.

Part I comprises the first two chapters. It is based on joint work with Professor Peter Z. G. Qian, Dr. Shifeng Xiong and Professor C. F. Jeff Wu. The joint paper is near submission for *Statistica Sinica*. In Chapter 1, a new class of designs called doubly sliced Latin hypercube design is proposed for running computer experiments. A DSLHD is a special LHD which can be partitioned into slices of smaller LHDs, each of which can be further partitioned into even smaller LHDs. In Section 1.2, I present two methods for constructing DSLHD. They are very flexible in the choice of run size and number of factors. In Section 1.3, the sampling properties of DSLHD are investigated. Definitions for the DSLH scheme, the SLH scheme and a few other schemes are given. It is shown that DSLHD is better for collective evaluation of computer simulations than LHD. A few examples are presented to illustrate the advantage of DSLHD for collective evaluation of different computer models in Section 1.4. In Section 1.5, it is shown that the DSLH scheme outperforms the SLH scheme for collective evaluation of computer models when we have more than one computer model and each of them has the same number of variants. The slicing idea is further generalized in Section 1.6, where we provide a more general construction of multi-layer sliced Latin hypercube designs. Both doubly sliced Latin hypercube designs and sliced Latin hypercube designs are special cases of multi-layer sliced Latin hypercube designs,

which correspond to two layers and one layer respectively.

In Chapter 2, we introduce a new procedure for batch sequential design based on DSLHD. The proposed procedure uses a slice of a DSLHD as the batch at each iteration. Since each slice of a DSLHD is an LHD, it is ensured that each batch has good space-filling properties. Batch sequential sampling are sometimes necessary because the turnaround time is shorter. I also incorporate some other criteria such as maximin distance criterion ([27]) to search for optimal batches at each iteration. The proposed procedure utilizes exchange algorithms to search for optimal batches within the DSLHD framework, which ensures the quality of the batches and is computationally efficient. A new criterion called the expected cross validation error is also proposed to make the sampling model-based. The new procedure is illustrated using some numerical examples and an interesting application to a data center example is presented. The numerical examples show that optimized sequential design is more efficient than randomized sequential design under the DSLHD framework.

Part II of the thesis comprises Chapters 3 and 4. It is based on joint work with Professor Xiaoming Huo and Professor Zhouwang Yang. One paper has been submitted to *Statistics and Probability Letters*. A second paper is near submission for *Annals of Statistics*. Smoothing or functional estimation is one of the most studied topics in statistics. It consists of many popular techniques as special cases, such as linear regression, smoothing splines, neural networks, etc. In this part of the thesis, I focus on the smoothing splines framework. A typical problem formulation in smoothing splines is to optimize an objective function which is a tradeoff between goodness-of-fit to the data and the roughness of the fitted function. The state-of-the art procedure to solve this problem is to assume the true function belongs to a functional class and convert the infinite dimensional functional estimation problem to a quadratic programming problem. This leads to well-known solutions such as natural cubic splines, thin plate splines, etc. However, this approach does not work well for irregular regions, as demonstrated by [52] and [72]. In this part of the thesis, we

propose a completely-data-driven smoothing approach to circumvent the irregular region problem. The main idea is to replace the penalty term by its estimate based on local least squares technique.

In Chapter 3, I focus on one-dimensional input and show that our new approach has exactly the same theoretical performance as the natural cubic splines. Specifically, our estimator achieves optimal convergence rate for nonparametric regression and achieves the sharp bound for minimax risk. All the derivations are based on the assumption of equally spaced design.

In Chapter 4, we derive more general theoretical properties of our estimator. With the same regularity conditions on the boundary as smoothing splines and some analytical assumptions on the underlying function, we show that our estimator enjoys all the nice theoretical properties that smoothing splines do. The derivation of the theoretical properties relies mainly on the connection between continuous semi-norms and discrete semi-norms in Sobolev space. More interestingly, we show through numerical experiments that our method is comparable to soap film smoothing ([72]) for irregular region smoothing and much better than thin plate splines. We also use a regular region smoothing example to show that our method works much better than soap film smoothing and is comparable to thin plate splines. It suggests that our method is essentially a local smoothing technique. This can circumvent the irregular region problem but also works for regular region problems.

Chapter 5 belongs to Part III of this thesis. It is based on joint work with Yeonsook Heo, Yuming Sun, Matthias H. Y. Tan, Professor Godfried Augenbroe and Professor C. F. Jeff Wu. It is part of a paper which was accepted by the 2011 building simulation conference and won the best student paper award in the conference. In this part, my focus is on uncertainty quantification of two microclimate parameters in building technology: (1) local wind speed, (2) wind pressure coefficient. We first use design of experiments to collect data for statistical analysis. Statistical models are then built to connect the standard model and

the meso-scale model. The explicit form of statistical models facilitate the improvement of standard models in the current simulation tools such as “EnergyPlus”.

CHAPTER I

DOUBLY SLICED LATIN HYPERCUBE DESIGNS (DSLHD)

1.1 Introduction

Computer experiments have become ubiquitous in science, engineering and services for studying complex phenomena. In scientific context, a computer experiment refers to mathematical modeling using computer simulation. In a computer simulation, a computer model replaces a traditional mathematical model. While a traditional mathematical model can usually be solved analytically, a computer model is usually solved numerically. Computer simulations can be divided into two types: stochastic simulations and deterministic simulations. Running a deterministic simulation again with the same inputs would yield the same output values ([58]; [19]). In this thesis, we only consider deterministic simulations.

A main goal in many computer experiments is to estimate the expected output of a computer model given a distribution of the inputs. To address this issue, [37] introduced Latin hypercube designs (LHD), referred to as *ordinary Latin hypercube designs* hereinafter. LHDs have since become very popular because of their good space-filling properties and the easy generation of such designs. In computer experiments, some inputs are qualitative, i.e., they take discrete values that can be either ordinal or non-ordinal. [50] presented a data center example which has seven inputs. Two of them are qualitative. In their paper, the main issue was how to model the effects of qualitative inputs. A related question is how to choose the input values at which to run the computer model. This has led to the introduction of *sliced Latin hypercube designs* (SLHD) in [47]. An SLHD is a special LHD that can be partitioned into slices of smaller LHDs. For a given set of values of the qualitative inputs, one can then assign one of the slices to the quantitative inputs. This arrangement has two attractive features: (1) The design for the quantitative inputs at

each combination of the qualitative inputs is an LHD. (2) The overall design for the quantitative inputs is an LHD. Since LHDs have good space-filling properties, SLHD makes the modelling more efficient in the presence of both qualitative and quantitative inputs. One can think of the computer model with specific values of the qualitative inputs as submodels. The evaluation of the computer model over different values of the qualitative and quantitative inputs can be thought of as collective evaluation of the submodels. More generally, it is desirable to use an SLHD to run a computer model in batches, with each batch of input values being one slice of the design. On the one hand, if it is feasible to borrow strength across the batches, the combined design should be an LHD. On the other hand, when data from different batches must be analyzed separately, the set of input values for each batch is guaranteed to be a smaller LHD. An earlier example of slicing in design construction is [73], which constructed central composite designs for quantitative and qualitative factors. An SLHD is also useful for running multiple computer models based on similar mathematics ([68]), where each model uses one slice of the design. In this chapter, we propose a generalized version of SLHD, called *doubly sliced Latin hypercube design* (DSLHD). A DSLHD is a special LHD that can be partitioned into smaller LHDs, each of which can be further partitioned into even smaller LHDs. Similar to SLHDs, for a given list of computer models, DSLHDs are useful for collective evaluation of all of them. DSLHDs can be more useful than SLHDs when we collectively evaluate more than one computer model and each of them has the same number of variants. For example, we can have three different computer models and each of them is solved using finite element method with different mesh densities. Details of the comparisons between DSLHDs and SLHDs are given in Section 1.5.

The remainder of the chapter is organized as follows. Section 1.2 presents two methods for constructing DSLHDs. The sampling properties of DSLHDs are derived in Section 1.3. Section 1.4 provides numerical illustration of the theoretical results in Section 1.3. Comparison of DSLHD and SLHD is given in Section 1.5. We then extend the slicing

structure to multi-layer sliced Latin hypercube design in Section 1.6. Summary and future research directions are presented in Section 1.7. Most of the technical proofs are in the appendix.

1.2 Construction of DSLHD

In this section, we present two methods for constructing DSLHDs. Both methods are easy to implement and can accommodate an arbitrary number of factors. The first method is easier to understand, while the second method can facilitate the derivation of the sampling properties of DSLHDs.

We begin with some definitions and notation. For positive integers m, s, t, and d, a DSLHD D associated with these parameters is an LHD with mst levels in d dimensions. Moreover, $\mathbf{D}$ can be partitioned into $\mathbf{D}_{ij}$'s as follows:

$$\begin{array}{cccc} \mathbf{D}_{11} & \mathbf{D}_{12} & \cdots & \mathbf{D}_{1t} \\ \mathbf{D}_{21} & \mathbf{D}_{22} & \cdots & \mathbf{D}_{2t} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{D}_{s1} & \mathbf{D}_{s2} & \cdots & \mathbf{D}_{st} \end{array},$$

where $\mathbf{D}_{ij}$ is a d-dimensional LHD with m levels for each $i = 1, \ldots, s,\ j = 1, \ldots, t$, $\mathbf{D}_i = \cup_{j=1}^{t} \mathbf{D}_{ij}$ is a d-dimensional LHD with mt levels for each $i = 1, \ldots, s$. Compared with an SLHD, each slice $\mathbf{D}_i$ in the DSLHD can be further partitioned into t smaller LHDs. Note that, for $t = 1$, a DSLHD becomes an SLHD. Throughout the paper, we use DSLHD(m, s, t, d) to denote a DSLHD associated with parameters m, s, t and d. Without loss of generality, the construction of DSLHD is done for d continuous factors, each of which takes values in $(0, 1]$.

A DSLHD(m, s, t, d) can be generated via the first method as follows.

Step 1. For $r = 1, \ldots, d$, do the following:

Construct the $mt \times s$ matrix $\mathbf{H} = (h_{\mu,\nu})$, whose ath row is a uniform permutation on $((a-1)s + 1, \ldots, as)$.

For $i = 1, \ldots, s$,

Construct the $m \times t$ matrix $\mathbf{C}^0 = (c^0_{\mu\nu})$, whose ath row is a uniform permutation on $(h_{(a-1)t+1,i}, \ldots, h_{at,i})$.

Construct the $m \times t$ matrix $\mathbf{C}^{ir} = (c^{ir}_{\mu\nu})$, whose bth column is a uniform permutation on the bth column of $\mathbf{C}^0$.

End the i loop.

End the r loop.

Step 2. For $i = 1, \ldots, s,\ j = 1, \ldots, t$, construct the $m \times d$ matrix $\mathbf{A}_{ij} = (a^{ij}_{\mu\nu})$, whose rth column is the jth column of $\mathbf{C}^{ir}$. These $\mathbf{A}_{ij}$'s are called the permutation matrices for the corresponding $\mathbf{D}_{ij}$'s.

Step 3. For $i = 1, \ldots, s,\ j = 1, \ldots, t$, construct the $m \times d$ matrix $\mathbf{D}_{ij}$, the (μ, ν)th entry of which is

$$x_{\mu\nu} = \frac{a^{ij}_{\mu\nu} - u^{ij}_{\mu\nu}}{n}, \quad \text{for } \mu = 1, \ldots, m,\ \nu = 1, \ldots, d, \tag{1.2.1}$$

where the $u^{ij}_{\mu\nu}$s are i.i.d. $U[0, 1)$ random variables, and $a^{ij}_{\mu\nu}$ and $u^{ij}_{\mu\nu}$ are mutually independent.

More definitions and notation are needed for introducing the second method. Suppose $\mathbf{A} = (a_{i,j,k})$ is an $m \times s \times t$ three-dimensional matrix. For $i = 1, \ldots, m,\ j = 1, \ldots, s,\ k = 1, \ldots, t$, let $\mathbf{A}(i, j, k) = a_{i,j,k}$, $\mathbf{A}(:, j, k) = (a_{1,j,k}, \ldots, a_{m,j,k})$, $\mathbf{A}(i, :, k) = (a_{i,1,k}, \ldots, a_{i,s,k})$, and $\mathbf{A}(i, j, :) = (a_{i,j,1}, \ldots, a_{i,j,t})$. For $a \in \mathfrak{R}$, let $\lceil a \rceil$ denote the smallest integer larger than or equal to a and $\lfloor a \rfloor$ the largest integer less than or equal to a. Note $\lceil x \rceil$ and $\lfloor x \rfloor$ are called the ceiling function and the floor function respectively. For a three-dimensional matrix $\mathbf{D} = (d_{i,j,k})$, let $\lfloor \mathbf{D} \rfloor = (\lfloor d_{i,j,k} \rfloor)$ and $\lceil \mathbf{D} \rceil = (\lceil d_{i,j,k} \rceil)$. The application of the ceiling function and floor function on a vector or a two-dimensional matrix is similarly defined, i.e., the two functions are applied to each element of the vector or the matrix. Let m, s, t be positive integers with $n = mst$. For an integer $b \geq 1$, let $\mathbf{Z}_b$ denote the set $\{1, \ldots, b\}$. Define a

permutation matrix PM(m, s, t) on $\mathbf{Z}_n$ to be an $m \times s \times t$ matrix in which each element of $\mathbf{Z}_n$ appears once. Suppose $\mathbf{A}$ is a PM(m, s, t). We call $\mathbf{A}$ an $m \times s \times t$ *doubly sliced permutation matrix*, denoted by DSPM(m, s, t), if $\mathbf{A}$ satisfies the following two conditions:

Condition 1. For $j = 1, \ldots, s$, $k = 1, \ldots, t$, $\lceil \mathbf{A}(:, j, k)/st \rceil$ forms a permutation on $\mathbf{Z}_m$.

Condition 2. For $j = 1, \ldots, s$, $\lceil \mathbf{A}(:, j, :)/s \rceil$ forms a permutation on $\mathbf{Z}_{mt}$.

A DSPM(m, s, t) can be generated in the following two steps.

Step 1. Construct an $mt \times s$ matrix $\mathbf{H} = (h_{\mu,\nu})$ whose ath row is a uniform permutation on $((a-1)s + 1, \ldots, as)$.

Step 2. For $i = 1, \ldots, s$, construct the $m \times t$ matrix $\mathbf{C}^0 = (c^0_{\mu\nu})$, whose ath row is a uniform permutation on $(h_{(a-1)t+1,i}, \ldots, h_{at,i})$. Randomly permutate each column of $\mathbf{C}^0$ with the permutations done for each column independently. Let $\mathbf{A}$ be an $m \times s \times t$ three-dimensional matrix with $\mathbf{A}(:, i, j)$ equal to the jth column of $\mathbf{C}^0$. Then $\mathbf{A}$ is the desired DSPM(m, s, t).

We now discuss how to use multiple doubly sliced permutation matrices to obtain a DSLHD. First generate d independent versions of DSPM(m, s, t), denoted by $\mathbf{H}_1, \ldots, \mathbf{H}_d$. For $r = 1, \ldots, s$, $c = 1, \ldots, t$, obtain an $m \times d$ matrix $\mathbf{A}_{rc}$ by letting its jth column be $\mathbf{H}_j(:, r, c)$. Obtain the following four-dimensional matrix

$$\mathbf{A}(:, r, c, i) = \mathbf{A}_{rc}(:, i), \tag{1.2.2}$$

where i is from 1 to d, and d is the dimension of the design variable. Construct an $n \times d$ matrix $\mathbf{B} = (b_{ki})$ such that $\mathbf{B}(:, i) = (\mathbf{A}(:, 1, 1, i), \mathbf{A}(:, 1, 2, i), \ldots, \mathbf{A}(:, 1, t, i), \ldots, \mathbf{A}(:, s, 1, i), \ldots, \mathbf{A}(:, s, t, i))^T$. An $n \times d$ design $\mathbf{D} = (d_{ki})$ is generated through

$$d_{ki} = (b_{ki} - u_{ki})/n, \tag{1.2.3}$$

where $k = 1, \ldots, n$, and $i = 1, \ldots, d$. u_{ki}'s are independent $U[0, 1)$ random variables, d_{ki} is the level of factor i on the kth run, and u_{ki} and b_{ki} are mutually independent. For $r = 1, \ldots, s$, $c = 1, \ldots, t$, let $\mathbf{D}_{rc}$ be the subset of points in $\mathbf{D}$ corresponding to $\mathbf{A}(:, r, c, :)$, and $\mathbf{D}_r = \bigcup_{c=1}^{t} \mathbf{D}_{rc}$. The properties of the constructed design $\mathbf{D}$ are summarized in Proposition 1.2.1.

Proposition 1.2.1. *Let m, s, and t be positive integers with n=mst. For* $\mathbf{D}$, $\mathbf{D}_{rc}$*s and* $\mathbf{D}_r$*s constructed above, the following hold:*

(i). $\mathbf{D}$ *is a d-dimensional Latin hypercube design with n levels;*

(ii). $\{\mathbf{D}_r\}_{r=1}^{s}$ *form a partition of* $\mathbf{D}$ *and each* $\mathbf{D}_r$ *is a d-dimensional Latin hypercube design with mt levels;*

(iii). $\{\mathbf{D}_{rc}\}_{c=1}^{t}$ *form a partition of* $\mathbf{D}_r$ *and each* $\mathbf{D}_{rc}$ *is a d-dimensional Latin hypercube design with m levels for* $r = 1, \ldots, s$.

Proof. The results are all obvious from the construction of $\mathbf{D}$. □

This proposition says that, when $\mathbf{D}$ is projected onto each of the d factors, precisely one point falls within each of the n intervals defined by $(0, 1/n]$,$(1/n, 2/n]$,. . .,$((n-1)/n, 1]$, and exactly one of the m points of each $\mathbf{D}_{rc}$ falls within each of the m intervals defined by $(0, 1/m]$,$(1/m, 2/m]$,. . .,$((m-1)/m, 1]$, and exactly one of the mt points of each $\mathbf{D}_r$ falls within each of the mt intervals defined by $(0, 1/mt]$,$(1/mt, 2/mt]$,. . .,$((mt-1)/mt, 1]$.

Let $s = 3$, $t = 2$, $m = 3$ and $d = 2$. An example of permutation matrix for a DSLHD(3, 3, 2, 2) is as follows (in transpose):

$$\mathbf{A}^T = \begin{pmatrix} 3 & 10 & 15 & 9 & 6 & 16 & 12 & 14 & 5 & 1 & 17 & 8 & 18 & 11 & 2 & 7 & 13 & 4 \\ 2 & 10 & 15 & 9 & 18 & 6 & 16 & 12 & 1 & 7 & 4 & 14 & 11 & 13 & 5 & 8 & 3 & 17 \end{pmatrix}.$$

The $\mathbf{A}$ matrix can be partitioned into six permutation matrixes corresponding to the six

slices of the DSLHD as follows:

$$\mathbf{A}_{11} = \begin{pmatrix} 3 & 2 \\ 10 & 10 \\ 15 & 15 \end{pmatrix}, \quad \mathbf{A}_{12} = \begin{pmatrix} 9 & 9 \\ 6 & 18 \\ 16 & 6 \end{pmatrix},$$

$$\mathbf{A}_{21} = \begin{pmatrix} 12 & 16 \\ 14 & 12 \\ 5 & 1 \end{pmatrix}, \quad \mathbf{A}_{22} = \begin{pmatrix} 1 & 7 \\ 17 & 4 \\ 8 & 14 \end{pmatrix},$$

$$\mathbf{A}_{31} = \begin{pmatrix} 18 & 11 \\ 11 & 13 \\ 2 & 5 \end{pmatrix}, \quad \mathbf{A}_{32} = \begin{pmatrix} 7 & 8 \\ 13 & 3 \\ 4 & 17 \end{pmatrix}.$$

Note that $\lceil \mathbf{A}_{ij}(:,k)/6 \rceil$ is a permutation on $\mathbf{Z}_3$ for $i = 1,2,3$, $j = 1,2$, and $k = 1,2$. Let $\mathbf{A}_i = \mathbf{A}_{i1} \bigcup \mathbf{A}_{i2}$. Then $\lceil \mathbf{A}_i(:,k)/3 \rceil$ is a permutation on $\mathbf{Z}_6$.

1.3 Sampling properties

In this section, we derive some sampling properties of DSLHDs using the following motivating example. Suppose there are t computer models, $f^{(1)}$, ..., $f^{(t)}$. Assume each $f^{(c)}$ has factors $\mathbf{x} = (x_1, \ldots, x_d)$, whose distribution is the uniform measure on $(0,1]^d$, denoted by F. For $c = 1, \ldots, t$, define $\mu_c = E[f^{(c)}(\mathbf{x})]$. For $c_1, c_2 = 1, \ldots, t$, define $cov_{c_1 c_2} = cov[f^{(c_1)}(\mathbf{x}), f^{(c_2)}(\mathbf{x})]$, which becomes $\sigma^2_{c_1} = var[f^{(c_1)}(\mathbf{x})]$ if $c_1 = c_2$. The goal here is to run each of $f^{(1)}, \ldots, f^{(t)}$ at m selected input values for the purpose of estimating $\mu_1, \ldots, \mu_t$. For $0 \le \lambda_c \le 1$, $c = 1, \ldots, t$, a linear combination of $\mu_1, \ldots, \mu_t$ given by

$$\eta = \sum_{c=1}^{t} \lambda_c \mu_c \tag{1.3.1}$$

is of interest in practice.

We consider five different schemes defined below to achieve this goal.

Definition 1. *Suppose m, t, t_1, and t_2 are positive integers with $n = mt = mt_1t_2$.*

(i). Let IID denote the scheme that takes an independent and identically distributed sample of m runs for each $f^{(c)}$, with the t samples generated independently.

(ii). Let LH denote the scheme that obtains t independent ordinary Latin hypercube designs of m runs, each of which is associated with one $f^{(c)}$.

(iii). Let SLH denote the scheme that produces an $n \times d$ SLHD with t slices by using the method in [47], where each slice is a smaller LHD with m levels and is assigned to one $f^{(c)}$.

(iv). Let DSLH denote the scheme that produces an $n \times q$ DSLHD with t_1 rows, each of which has t_2 slices and each slice has m levels. Each slice of the generated DSLHD is assigned to one $f^{(c)}$.

(v). Let SPLIT denote the scheme that randomly splits an ordinary LHD of n runs into t subsets of equal size and each subset is assigned to one $f^{(c)}$.

Expectation, variance and covariance under these schemes (in the same order) are denoted by the subscripts IID, LH, SLH, DSLH, and SPLIT, respectively. The SLH scheme is referred to as the *sliced Latin hypercube sampling*. The DSLH scheme is referred to as the *doubly sliced Latin hypercube sampling*. For any of these schemes, let $\mathbf{D}^{(c)}$ denote the design set for $f^{(c)}$, $c = 1, \ldots, t$. Denote by $\mathbf{d}_i^{(c)}$ the ith row of $\mathbf{D}^{(c)}$ and $d_{ik}^{(c)}$ the (i,k)th entry of $\mathbf{D}^{(c)}$. Obtain a design $\mathbf{D}$ by combining all rows of $\mathbf{D}^{(1)}$, ..., $\mathbf{D}^{(t)}$. For $c = 1, \ldots, t$, μ_c is estimated by

$$\hat{\mu}_c = m^{-1} \sum_{i=1}^{m} f^{(c)}(\mathbf{d}_i^{(c)}), \tag{1.3.2}$$

and η by

$$\hat{\eta} = \sum_{c=1}^{t} \lambda_c \hat{\mu}_c. \tag{1.3.3}$$

Remark 1. *When the t models are the same, i.e., $f^{(c)} = f$ and $\lambda_c = t^{-1}$, for $c = 1, \ldots, t$, we have $\eta = E[f(x)]$. Then the collective evaluation of the t models becomes evaluation of a single computer model.*

For later development, we describe the ANOVA decomposition of integrable functions on $[0,1]^d$ ([43]; [35]). Recall that F is the uniform measure on $(0,1]^d$. Let $dF = \prod_{k=1}^{d} dF_k$ and $dF_{-k} = \prod_{l \neq k} dF_l$. If $f{:}R^d \to R$ is a measurable function of $\mathbf{x} = (x_1, \ldots, x_d)$ and $E[f(\mathbf{x})]^2$ is well defined and finite, then f can be decomposed as

$$f(\mathbf{x}) = \mu + \sum_{k=1}^{d} f_{-k}(x_k) + r(\mathbf{x}), \tag{1.3.4}$$

where μ is the grand mean given by $\int f(\mathbf{x}) dF$ and the functional main effect of x_k is

$$f_{-k}(x_k) = \int [f(\mathbf{x}) - \mu] dF_{-k}. \tag{1.3.5}$$

For $k = 1, \ldots, d$, note that $\int f_{-k} dF_k = 0$ and

$$\int r(\mathbf{x}) dF_{-k} = 0. \tag{1.3.6}$$

Next, we present some results on the joint probability mass functions of the permutation matrices constructed in Section 1.2.

Lemma 1.3.1. *For positive integers m, s, t with $n = mst$, let $\mathbf{H}$ be a DSPM(m, s, t) on $\mathbf{Z}_n$ generated by using the second method in Section 1.2. Denote by h_{ijk} the (i, j, k)th entry of $\mathbf{H}$. Consider $u, v, w \in \mathbf{Z}_n$.*

(i). *For $i = 1, \ldots, m$, $j = 1, \ldots, s$, $k = 1, \ldots, t$, the probability mass function for h_{ijk} is*

$$Pr(h_{ijk} = u) = 1/n. \tag{1.3.7}$$

(ii). *For $i_1, i_2 = 1, \ldots, m$, $i_1 \neq i_2$ and $j = 1, \ldots, s$, $k = 1, \ldots, t$, the joint probability mass function for $h_{i_1 jk}$ and $h_{i_2 jk}$ is*

$$Pr(h_{i_1jk} = u, h_{i_2jk} = v) = \begin{cases} [n(n-st)]^{-1}, & \lceil u/st \rceil \neq \lceil v/st \rceil, \\ 0, & otherwise. \end{cases} \tag{1.3.8}$$

(iii). *For i_1, $i_2 = 1, \ldots, m$, $j = 1, \ldots, s$, k_1, $k_2 = 1, \ldots, t$, $k_1 \neq k_2$, the joint probability mass function for $h_{i_1jk_1}$ and $h_{i_2jk_2}$ is*

$$Pr(h_{i_1jk_1} = u, h_{i_2jk_2} = v) = \begin{cases} [n(n-sm)]^{-1}, & \lceil u/st \rceil = \lceil v/st \rceil, \lceil (u - \lfloor u/st \rfloor)/s \rceil \neq \lceil (v - \lfloor v/st \rfloor)/s \rceil, \\ 1/n^2, & \lceil u/st \rceil \neq \lceil v/st \rceil, \\ 0, & otherwise. \end{cases} \tag{1.3.9}$$

(iv). *For i_1, $i_2 = 1, \ldots, m$, j_1, $j_2 = 1, \ldots, s$, $j_1 \neq j_2$, k_1, $k_2 = 1, \ldots, t$, the joint probability mass function for $h_{i_1j_1k_1}$ and $h_{i_2j_2k_2}$ is*

$$Pr(h_{i_1j_1k_1} = u, h_{i_2j_2k_2} = v) = \begin{cases} 1/n^2, & \lceil u/st \rceil \neq \lceil v/st \rceil, \\ [n(n-mt)]^{-1}, & \lceil u/st \rceil = \lceil v/st \rceil, \lceil (u - \lfloor u/st \rfloor)/s \rceil = \lceil (v - \lfloor v/st \rfloor)/s \rceil, \\ 1/n^2, & \lceil u/st \rceil = \lceil v/st \rceil, \lceil (u - \lfloor u/st \rfloor)/s \rceil \neq \lceil (v - \lfloor v/st \rfloor)/s \rceil, \\ 0, & otherwise. \end{cases} \tag{1.3.10}$$

These probability mass functions are more complicated than those of a uniform permutation on $\mathbf{Z}_n$ used in the construction of an ordinary LHD. Also notice that when $t = 1$, (ii) and (iv) reduce to (ii) and (iii) of Lemma 1 in [47] respectively. Thus our Lemma 1.3.1 can be viewed as an extension of Lemma 1 in [47].

Let $\mathbf{D}$ be an $n \times d$ DSLHD with parameters m, s, t and d, and $\mathbf{D}_{rc}$ is defined as before, where $r = 1, \ldots, s$, and $c = 1, \ldots, t$. The sampling distribution of $\mathbf{D}_{rc}$ is given in Lemma 1.3.2.

Lemma 1.3.2. *For positive integers m, s and t with $n = mst$, $\mathbf{D}_{rc}$ constructed in Section 1.2 is statistically equivalent to an $m \times d$ ordinary Latin hypercube design.*

In some cases, we are also interested in the sampling distributions of $\mathbf{D}_r$. The next lemma gives such a result.

Lemma 1.3.3. *For positive integers m, t and s with $n = mst$, $\mathbf{D}_r$ constructed in Section 1.2 is statistically equivalent to an SLHD of t slices, each of which is an ordinary Latin hypercube design with m levels.*

Next we present results on $\hat{\mu}_c$ in (1.3.2) and $\hat{\eta}$ in (1.3.3) under doubly sliced Latin hypercube sampling. The following theorem provides a finite sample result under some monotonicity assumptions on $f^{(1)},\ldots,f^{(t)}$.

Theorem 1.3.1. *Suppose that, for $c = 1,\ldots,t$, $f^{(c)}(\mathbf{x})$ is monotonic in each argument x_i of $\mathbf{x} = (x_1,\ldots,x_d)$ and any pair of functions $f^{(c_1)}$ and $f^{(c_2)}$, $c_1 \neq c_2$, is jointly increasing or decreasing in each argument x_i of $\mathbf{x}$. For the five schemes described in Definition 1, we have*

(i). *For $c = 1,\ldots,t$ and $\hat{\mu}_c$ defined in (1.3.2),*

$$var_{DSLH}(\hat{\mu}_c) \leq var_{IID}(\hat{\mu}_c). \tag{1.3.11}$$

(ii). *For $\hat{\eta}$ defined in (1.3.3),*

$$var_{DSLH}(\hat{\eta}) \leq var_{LH}(\hat{\eta}) \leq var_{IID}(\hat{\eta}). \tag{1.3.12}$$

By dropping the monotonicity assumptions in Theorem 1.3.1, we have a more general result for $\hat{\mu}_c$ and $\hat{\eta}$ under doubly sliced Latin hypercube sampling as follows.

Theorem 1.3.2. *Suppose that $E[f^{(1)}(\mathbf{x})]^2,\ldots,E[f^{(t)}(\mathbf{x})]^2$ are all well defined and finite. For $c = 1,\ldots,t$, let $f^{(c)}_{-k}$ be the functional main effect for the variable x_k of $\mathbf{x} = (x_1,\ldots,x_d)$ in the ANOVA decomposition of $f^{(c)}$ in (1.3.5). Let m, t, t_1 and t_2 be positive integers with $n = mt$ and $t = t_1t_2$ and let the DSLH scheme and the SLH scheme be defined as in Definition 1. Then as $n \to \infty$ with t fixed, we have*

(i). *For $c = 1, \ldots, t$ and $\hat{\mu}_c$ defined in (1.3.2) based on the slice $D^{(c)}$,*

$$var_{DSLH}(\hat{\mu}_c) = var_{SLH}(\hat{\mu}_c) = \sigma_c^2 \frac{t}{n} - \frac{t}{n} \sum_{k=1}^{d} \int_0^1 [f_{-k}^{(c)}(x_k)]^2 dx_k + o(n^{-1}). \quad (1.3.13)$$

(ii). *For $\hat{\eta}$ defined in (1.3.3) associated with the slices $D^{(1)}, \ldots, D^{(t)}$,*

$$var_{DSLH}(\hat{\eta}) = \frac{t}{n} \sum_{c=1}^{t} \lambda_c^2 \sigma_c^2 - \frac{t}{n} \sum_{c=1}^{t} \{\lambda_c^2 \sum_{k=1}^{d} \int_0^1 [f_{-k}^{(c)}(x_k)]^2 dx_k\} + o(n^{-1}). \quad (1.3.14)$$

Some observations based on Theorem 1.3.2 are worth noting. First, in (i) and (ii) of the theorem, the main effects of $f_k^{(c)}(x_k)$ are filtered out, thus achieving the variance reduction similar to an ordinary LHD. Second, when all the $f^{(c)}$s are the same and $\lambda_c = t^{-1}$ as described in Remark 1, we have $\sigma^2 = \sigma_c^2 = var[f(\mathbf{x})]$ and the formula in (ii) is reduced to $var_{DSLH}(\hat{\eta}) = \frac{1}{n}\sigma^2 - \frac{1}{n}\sum_{k=1}^{d} \int_0^1 [f_{-k}(x_k)]^2 dx_k + o(n^{-1})$, which is similar to that of an SLHD of n runs as given in [47] and that of an ordinary LHD of n runs as given in [60] and [35]. Moreover, if the functions are different, the LH scheme, the SLH scheme and the DSLH scheme are asymptotically equivalent in the sense that $var_{LH}(\hat{\eta})$, $var_{SLH}(\hat{\eta})$, and $var_{DSLH}(\hat{\eta})$ all have the order $O(n^{-1})$.

1.4 Numerical illustration

In this section, we provide numerical illustrations to corroborate some theoretical results in the previous section.

Example 1.1. This example uses a five-dimensional function ([16])

$$f(\mathbf{x}) = log(x_1 x_2 x_3 x_4 x_5). \quad (1.4.1)$$

Note we use only one computer model here. This is the case described in Remark 1. Consider using four batches of runs to estimate $\eta = E[f(\mathbf{x})]$, where the distribution of $\mathbf{x}$ is the uniform measure on $(0, 1]^5$ and $\mathbf{D}^{(1)}, \ldots, \mathbf{D}^{(4)}$ are the sets of input values in the four batches respectively. The size of the four batches is the same and is denoted by m. We estimate η by $\hat{\eta}$ based on $\mathbf{D}^{(1)}, \ldots, \mathbf{D}^{(4)}$, defined in (1.3.3) with $\lambda_1 = \lambda_2 = \lambda_3 = \lambda_4 = 1/4$ and

$\lambda_1 = 1/2$, $\lambda_2 = \lambda_3 = \lambda_4 = 1/6$, respectively; we estimate μ_1 by $\hat{\mu}_1$ based on $\mathbf{D}^{(1)}$, defined in (1.3.2). Compare the five schemes (IID, LH, SLH, DSLH, SPLIT) to produce $\mathbf{D}^{(1)}, \ldots, \mathbf{D}^{(4)}$. For IID, we generate four independent and identically distributed samples of m runs. For LH, we generate four independent ordinary LHDs with m levels. For SLH, we generate an SLHD of four slices, each of which is a smaller ordinary LHD with m levels. For DSLH, we generate a DSLHD(m, s, 4, 5) and use the four slices in the first row as $\mathbf{D}^{(1)}, \ldots, \mathbf{D}^{(4)}$. Note that, unlike in Definition 1, here we only use the first row of a generated DSLHD for the DSLH scheme whose sole purpose is to corroborate the results in Lemmas 1.3.2 and 1.3.3. Also s can be any integer greater than one and does not affect the numerical results. For SPLIT, we generate an ordinary LHD of $4m$ runs and randomly split them into four subsets of equal size.

For each scheme, we compute $\hat{\eta}$ and $\hat{\mu}_1$ 2000 times for $m = 5, 10, 20, 40$. Table 1.4.1 presents the root mean square error (RMSE) of $\hat{\mu}_1$ over the replicates for each scheme. The RMSEs for $\hat{\eta}$ with $\lambda_1 = \lambda_2 = \lambda_3 = \lambda_4 = 1/4$ and $\lambda_1 = 1/2$, $\lambda_2 = \lambda_3 = \lambda_4 = 1/6$ are given in Tables 1.4.2 and 1.4.3 respectively.

Table 1.4.1: RMSEs of $\hat{\mu}_1$ in Example 1.1

	IID	SPLIT	LH	SLH	DSLH
$m = 5$	1.0037	0.9096	0.4741	0.4498	0.4662
$m = 10$	0.7105	0.6260	0.2329	0.2371	0.2322
$m = 20$	0.4959	0.4403	0.1158	0.1180	0.1156
$m = 40$	0.3480	0.2993	0.0589	0.0566	0.0573

Table 1.4.2: RMSEs of $\hat{\eta}$ in Example 1.1 with $\lambda_1 = \lambda_2 = \lambda_3 = \lambda_4 = 1/4$

	IID	SPLIT	LH	SLH	DSLH
$m = 5$	0.4917	0.1189	0.2458	0.1137	0.1138
$m = 10$	0.3597	0.0577	0.1163	0.0584	0.0576
$m = 20$	0.2563	0.0296	0.0577	0.0295	0.0292
$m = 40$	0.1755	0.0145	0.0297	0.0148	0.0144

Table 1.4.3: RMSEs of $\hat{\eta}$ in Example 1.1 with $\lambda_1 = 1/2$ and $\lambda_2 = \lambda_3 = \lambda_4 = 1/6$

	IID	SPLIT	LH	SLH	DSLH
$m = 5$	0.5716	0.3196	0.2857	0.1879	0.1882
$m = 10$	0.4071	0.2167	0.1356	0.0961	0.0942
$m = 20$	0.2962	0.1467	0.0654	0.0484	0.0461
$m = 40$	0.2105	0.1008	0.0342	0.0243	0.0226

Some observations from Tables 1.4.1, 1.4.2, and 1.4.3 are noted as follows:

(i). As expected, IID performs the worst throughout the simulations.

(ii). Among the RMSEs of $\hat{\mu}_1$ shown in Table 1.4.1, LH, SLH and DSLH have similar performance. This is because a slice in an SLHD or a DSLHD is statistically equivalent to an ordinary LHD of equal size.

(iii). For the RMSEs of $\hat{\eta}$, SLH and DSLH consistently outperform LH because for each argument of the design variable, any pair of inputs from two different slices of an SLHD or a DSLHD is negatively correlated and the single function serving as the computer model is increasing in each argument of the design variable. The results then follow from Theorem 1.3.1 since its conditions are satisfied.

(iv). When the λ_is are equal, the performances of SPLIT, SLH and DSLH are similar. However, when the λ_is are unequal, SLH and DSLH outperform SPLIT. In both situations, DSLH performs similarly to SLH because a row in a DSLHD is statistically equivalent to an SLHD.

Next we will explain why the λ_is have an effect on the comparisons between SPLIT, SLH and DSLH. Since DSLH and SLH perform similarly, we focus on the comparisons between SPLIT and SLH. The first question is: when the λ_is are equal, why do SPLIT and SLH perform similarly? We claim that, when there is only one single computer model and the λ_is are equal, SPLIT is equivalent to using an ordinary LHD as the input values for the

estimation of the mean of a computer model. This can be proved as follows. Use τ to denote the permutation in the SPLIT scheme after an ordinary LHD is generated. For the expectation of the estimator under the SPLIT scheme, the following holds:

$$E_{SPLIT}(\hat{\eta}) = E_{\tau}(E_{SPLIT|\tau}(\hat{\eta})) = E_{\tau}(E_{LHD}(\hat{\eta})) = E_{LHD}(\hat{\eta}), \tag{1.4.2}$$

where the last equality holds because the permutation τ does not change the average of the sampled f values.

For the variance of the estimator under the SPLIT scheme, we have:

$$\begin{aligned} var_{SPLIT}(\hat{\eta}) &= E_{\tau}(var_{SPLIT|\tau}(\hat{\eta})) + var_{\tau}(E_{SPLIT|\tau}(\hat{\eta})) \\ &= E_{\tau}(var_{LHD}(\hat{\eta})) + var_{\tau}(E_{LHD}(\hat{\eta})) \\ &= var_{LHD}(\hat{\eta}). \end{aligned} \tag{1.4.3}$$

We thus prove the equivalence of the SPLIT scheme and an ordinary LHD when used for estimating the mean of a computer model.

We now explain why, when the λ_is are equal, the estimator of η based on an ordinary LHD and on an SLHD have similar RMSEs. We first provide a useful result in the following lemma.

Lemma 1.4.1. *Given $f(x) : \mathbf{R} \to \mathbf{R}$, where x is uniformly distributed on $(0, 1]$, let μ be the mean of $f(x)$. For positive integers m, t, and n such that $n = mt$, let $x_1^{LHD}, \ldots, x_n^{LHD}$ denote the input values from an ordinary LHD of n runs, $x_1^{SLHD}, \ldots, x_n^{SLHD}$ denote the input values from an SLHD of t slices, each of which has m runs. Let $\hat{\mu}^{LHD} = \frac{1}{n}\sum_{i=1}^{n} f(x_i^{LHD})$ and $\hat{\mu}^{SLHD} = \frac{1}{n}\sum_{i=1}^{n} f(x_i^{SLHD})$. Then $E(\hat{\mu}^{LHD} - \mu)^2 = E(\hat{\mu}^{SLHD} - \mu)^2$.*

Based on the result in Lemma 1.4.1, we provide a more general result in the following proposition.

Proposition 1.4.1. *Given $f(\mathbf{x}) : \mathbf{R}^d \to \mathbf{R}$, where $\mathbf{x}$ is uniformly distributed on $(0, 1]^d$, let μ be the mean of $f(\mathbf{x})$. Suppose $f(\mathbf{x})$ is additive, i.e., $f(\mathbf{x}) = f_1(x_1) + \ldots + f_d(x_d)$, where x_i is*

the ith component of $\mathbf{x}$. *For positive integers m, t, and n such that* $n = mt$, *let* $\mathbf{x}_1^{LHD}, \ldots, \mathbf{x}_n^{LHD}$ *denote the input values from an ordinary LHD of n runs,* $\mathbf{x}_1^{SLHD}, \ldots, \mathbf{x}_n^{SLHD}$ *denote the input values from an SLHD of t slices, each of which has m runs. Let* $\mu^{LHD} = \frac{1}{n}\sum_{i=1}^{n} f(\mathbf{x}_i^{LHD})$ *and* $\mu^{SLHD} = \frac{1}{n}\sum_{i=1}^{n} f(\mathbf{x}_i^{SLHD})$. *Then* $E(\hat{\mu}^{LHD} - \mu)^2 = E(\hat{\mu}^{SLHD} - \mu)^2$.

Proposition 1.4.1 says that, when $f(\mathbf{x})$ is additive, the sample average estimator for $E(f(\mathbf{x}))$ based on an ordinary LHD and that on an SLHD of the same size are unbiased and have the same variance. In fact, this can be extended to DSLHD as stated in the following corollary.

Corollary 1.4.1. *Given* $f(\mathbf{x}) : \mathbf{R}^d \to \mathbf{R}$, *where* $\mathbf{x}$ *is uniformly distributed on* $(0, 1]^d$, *let* μ *be the mean of* $f(\mathbf{x})$. *Suppose* $f(\mathbf{x})$ *is additive, i.e.,* $f(\mathbf{x}) = f_1(x_1) + \ldots + f_d(x_d)$, *where* x_i *is the ith component of* $\mathbf{x}$. *For positive integers m, t,* t_1, t_2, *and n such that* $n = mt = mt_1t_2$, *let* $\mathbf{x}_1^{LHD}, \ldots, \mathbf{x}_n^{LHD}$ *denote the input values from an ordinary LHD of n runs,* $\mathbf{x}_1^{SLHD}, \ldots, \mathbf{x}_n^{SLHD}$ *denote the input values from an SLHD of t slices, each of which has m runs, and* $\mathbf{x}_1^{DSLHD}, \ldots, \mathbf{x}_n^{DSLHD}$ *denote the input values from a DSLHD(m,* t_1, t_2, *d). Let* $\mu^{LHD} = \frac{1}{n}\sum_{i=1}^{n} f(\mathbf{x}_i^{LHD})$, $\mu^{SLHD} = \frac{1}{n}\sum_{i=1}^{n} f(\mathbf{x}_i^{SLHD})$, *and* $\mu^{DSLHD} = \frac{1}{n}\sum_{i=1}^{n} f(\mathbf{x}_i^{DSLHD})$. *Then* $E(\hat{\mu}^{LHD} - \mu)^2 = E(\hat{\mu}^{SLHD} - \mu)^2 = E(\hat{\mu}^{DSLHD} - \mu)^2$.

Proof. The proof is similar to that of Proposition 1.4.1 and thus omitted. □

Since the test function in Example 1 is additive, from Proposition 1.4.1, we know the estimator of the SPLIT scheme and that of the SLH scheme have the same RMSE when the λ_is are the same. Now let us explain why when the λ_is are different, SLH and DSLH outperform SPLIT. Let $\hat{\mu}_1^{SPLIT}, \ldots, \hat{\mu}_4^{SPLIT}$ be the four estimators corresponding to the $\mathbf{D}^{(i)}$s from the SPLIT scheme and let $\hat{\mu}_1^{SLH}, \ldots, \hat{\mu}_4^{SLH}$ denote the four estimators corresponding to the $\mathbf{D}^{(i)}$s from the SLH scheme. Since $var_{SPLIT}(\hat{\eta}) = var_{SLH}(\hat{\eta})$, for equal λ_is, we have

$$4var_{SPLIT}(\hat{\mu}_1^{SPLIT}) + 6cov_{SPLIT}(\hat{\mu}_1^{SPLIT}, \hat{\mu}_2^{SPLIT}) = 4var_{SLH}(\hat{\mu}_1^{SLH}) + 6cov_{SLH}(\hat{\mu}_1^{SLH}, \hat{\mu}_2^{SLH}). \tag{1.4.4}$$

We conjecture that

$$var_{SPLIT}(\hat{\mu}_1^{SPLIT}) > var_{SLH}(\hat{\mu}_1^{SLH}). \tag{1.4.5}$$

Heuristically, this is because $\mathbf{D}^{(1)}$ in the SLH scheme is an ordinary LHD, while $\mathbf{D}^{(1)}$ in the SPLIT scheme is a random subset of an ordinary LHD. Based on the conjecture, we have

$$cov_{SPLIT}(\hat{\mu}_1^{SPLIT}, \hat{\mu}_2^{SPLIT}) < cov_{SLH}(\hat{\mu}_1^{SLH}, \hat{\mu}_2^{SLH}) < 0. \tag{1.4.6}$$

We are now ready to explain the effect of the λ_is on $var_{SPLIT}(\hat{\eta})$ and $var_{SLH}(\hat{\eta})$. For SPLIT, we can calculate $var_{SPLIT}(\hat{\eta})$ as follows:

$$\begin{aligned} var_{SPLIT}(\hat{\eta}) &= \sum_{i=1}^{4} \lambda_i^2 var_{SPLIT}(\hat{\mu}_i^{SPLIT}) + \sum_{i,j=1,i\neq j}^{4} \lambda_i \lambda_j cov_{SPLIT}(\hat{\mu}_i^{SPLIT}, \hat{\mu}_j^{SPLIT}) \\ &= var_{SPLIT}(\hat{\mu}_1^{SPLIT}) \sum_{i=1}^{4} \lambda_i^2 + cov_{SPLIT}(\hat{\mu}_1^{SPLIT}, \hat{\mu}_2^{SPLIT}) \sum_{i,j=1,i\neq j}^{4} \lambda_i \lambda_j \end{aligned} \tag{1.4.7}$$

Since $\sum_{i=1}^{4} \lambda_i^2$ is minimized and $\sum_{i,j=1,i\neq j}^{4} \lambda_i \lambda_j$ is maximized when the λ_is are equal, we know that the two terms in (1.4.7) both increase when the λ_is change from being equal to unequal. Similarly, the change of the λ_is from being equal to unequal also increases $var_{SLH}(\hat{\eta})$. However, the amount of increase in $var_{SLH}(\hat{\eta})$ is smaller than that in $var_{SPLIT}(\hat{\eta})$ because of (1.4.5) and (1.4.6). This explains heuristically why SLH and DSLH outperform SPLIT when the λ_is are different.

Next we use the following example to compare the five schemes in Definition 1.

Example 1.2. This example uses the following functions:

$$\begin{aligned} f_1(\mathbf{x}) &= log(\frac{1}{\sqrt{x_1}} + \frac{1}{\sqrt{x_2}}), \\ f_2(\mathbf{x}) &= log(\frac{1}{x_1} + \frac{1}{x_2}), \\ f_3(\mathbf{x}) &= -x_1 - x_2, \\ f_4(\mathbf{x}) &= -x_1^2 - x_2^2. \end{aligned}$$

to act as four different computer models. The distribution of $\mathbf{x}$ is the uniform measure on $(0, 1]^2$. Note, unlike Example 1, the DSLH scheme is the same as defined in Definition 1.

Specifically, for the DSLH scheme, we generate a DSLHD(m, 2, 2, 2) and assign the two slices in the first row to f_1 and f_2, and the two slices in the second row to f_3 and f_4. For each of the five schemes in Definition 1, we computed $\hat{\mu}_1$, $\hat{\eta}$ with $\lambda_1 = \lambda_2 = \lambda_3 = \lambda_4 = 1/4$ and $\lambda_1 = 1/2$, $\lambda_2 = \lambda_3 = \lambda_4 = 1/6$ 2000 times for $m = 5, 10, 20, 40$. Tables 1.4.4, 1.4.5, and 1.4.6 present the RMSEs of $\hat{\eta}$ and $\hat{\mu}_1$ over the 2000 replicates.

Table 1.4.4: RMSEs of $\hat{\mu}_1$ in Example 1.2

	IID	SPLIT	LH	SLH	DSLH
$m = 5$	0.2011	0.1768	0.1182	0.1081	0.1103
$m = 10$	0.1370	0.1204	0.0618	0.0620	0.0601
$m = 20$	0.0969	0.0860	0.0330	0.0327	0.0328
$m = 40$	0.0698	0.0606	0.0179	0.0184	0.0173

Table 1.4.5: RMSEs of $\hat{\eta}$ in Example 1.2 with $\lambda_1 = \lambda_2 = \lambda_3 = \lambda_4 = 1/4$

	IID	SPLIT	LH	SLH	DSLH
$m = 5$	0.1320	0.0802	0.0756	0.0645	0.0599
$m = 10$	0.0933	0.0542	0.0393	0.0349	0.0328
$m = 20$	0.0654	0.0365	0.0214	0.0200	0.0182
$m = 40$	0.0481	0.0261	0.0117	0.0110	0.0101

Table 1.4.6: RMSEs of $\hat{\eta}$ in Example 1.2 with $\lambda_1 = 1/2$ and $\lambda_2 = \lambda_3 = \lambda_4 = 1/6$

	IID	SPLIT	LH	SLH	DSLH
$m = 5$	0.1292	0.0790	0.0793	0.0624	0.0546
$m = 10$	0.0909	0.0531	0.0396	0.0340	0.0307
$m = 20$	0.0639	0.0354	0.0216	0.0189	0.0172
$m = 40$	0.0451	0.0254	0.0114	0.0107	0.0098

Some interesting observations from Tables 1.4.4, 1.4.5, and 1.4.6 are given as follows:

(i). As in Example 1.1, IID is the worst in all situations.

(ii). For the estimation of μ_1, LH, SLH and DSLH perform similarly because a slice in an SLHD or a DSLHD is statistically equivalent to an ordinary LHD of the same size.

These three schemes are better than SPLIT because the subset used to estimate μ_1 in the SPLIT scheme is a random subset of an ordinary LHD rather than an ordinary LHD.

(iii). For the estimation of η, SLH and DSLH are better than SPLIT even if the λ_is are the same because there are four different functions instead of a single function. Because SLH and DSLH ensure that the subset assigned to each function is an ordinary LHD, they achieve the space-filling property in the design for each function. By contrast SPLIT does not have this property.

(iv). For the estimation of η, SLH and DSLH are better than LH. This follows from Theorem 1.3.1 (ii) since the conditions in Theorem 1.3.1 are satisfied by the four functions in Example 1.2.

(v). DSLH is slightly better than SLH. However, the finite sample comparison between DSLH and SLH is problem-dependent. In fact, as shown in Theorem 1.3.2, the asymptotic performances of SLH and DSLH are similar. We will compare these two schemes in the next section.

1.5 Comparisons between DSLHD and SLHD

In this section, we consider the comparisons between the SLH scheme and the DSLH scheme. In the previous section, we observed that the two schemes perform similarly for the estimation of both the mean of one computer model and a weighted average of the means of several computer models. The question is: when will the DSLH scheme be better than the SLH scheme? To answer this question, we first give a slightly different setup of the estimation problem.

Suppose there are s_2 computer models and each of them has s_1 variants. For example, we can have s_2 different finite element models and each of them is solved using s_1 different mesh densities. Denote the computer models as f_{ij}, $i = 1, \ldots, s_2$, $j = 1, \ldots, s_1$, where f_{ij} is

the jth variant of the ith computer model. Suppose each f_{ij} has factors $\mathbf{x} = (x_1, \ldots, x_d)$ with the uniform distribution F on $(0,1]^d$. For $c_2 = 1, \ldots, s_2$, $c_1 = 1, \ldots, s_1$, define $\mu_{c_2c_1} = \mathrm{E}[f_{c_2c_1}(\mathbf{x})]$. For c_{11}, $c_{12} = 1, \ldots, s_1$, $c_{21}, c_{22} = 1, \ldots, s_2$, define $\mathrm{cov}_{c_{21}c_{11}c_{22}c_{12}} = \mathrm{cov}[f_{c_{21}c_{11}}(\mathbf{x}), f_{c_{22}c_{12}}(\mathbf{x})]$, which becomes $\sigma^2_{c_{21}c_{11}} = \mathrm{var}[f_{c_{21}c_{11}}(\mathbf{x})]$ if $c_{21} = c_{22}$ and $c_{11} = c_{12}$. The goal here is to run each f_{ij} at m selected input values for the purpose of estimating μ_{ij}. For $0 \le \lambda_{ij} \le 1$, $i = 1, \ldots, s_2$, $j = 1, \ldots, s_1$, the following linear combinations are of interest in practice

$$\begin{aligned} \eta &= \sum_{i=1}^{s_2} \sum_{j=1}^{s_1} \lambda_{ij}\mu_{ij}, \\ \eta_i &= \sum_{j=1}^{s_1} \lambda_{ij}\mu_{ij}, \quad i = 1, \ldots, s_2. \end{aligned} \tag{1.5.1}$$

To compare DSLHD and SLHD as sampling methods for the aforementioned estimation problem, we give definition of three variants of the SLH scheme and the DSLH scheme as follows.

Definition 2. *Suppose m, s, s_1, and s_2 are positive integers with $n = ms = ms_1s_2$.*

(i). Let SLH1 denote the scheme that produces an $n \times d$ SLHD with s slices by using the method in [47], where each slice is a smaller LHD with m levels and is assigned to one f_{ij}.

(ii). Let SLH2 denote the scheme that independently produces s_2 $ms_1 \times d$ SLHDs, each of which has s_1 slices, by using the method in [47]. For $i = 1, \ldots, s_2$, $j = 1, \ldots, s_1$, assign the jth slice in the ith SLHD to f_{ij}.

(iii). Let SLH3 denote the scheme that generates an $n \times d$ SLHD with s_2 slices by using the method in [47], where each slice is a smaller LHD with ms_1 levels. For $i = 1, \ldots, s_2$, randomly split the ith slice into s_1 subsets of size m and assign each of the subsets to one variant of the ith computer model.

(iv). Let DSLH denote the scheme that produces a DSLHD(m,s_2,s_1,d). For $i = 1, \ldots, s_2$, $j = 1, \ldots, s_1$, D_{ij} is assigned to f_{ij}.

In Section 1.5.2, we will compare these four schemes on the aforementioned estimation problem. Before that, we present some theoretical results in Section 1.5.1 which will help to understand the implications of the numerical experiments in Section 1.5.2.

1.5.1 Some theoretical results

In this subsection, we explore the covariance structure of SLHD and DSLHD. This structure is essential to understand the difference between SLHD, DSLHD and ordinary LHD. We begin with the following proposition.

Proposition 1.5.1. *For an SLHD on* $(0, 1]$*, let* t *denote the number of slices, and* m *the number of runs in each slice, the covariance between any two points from two different slices is non-positive and increases as* t *increases with* m *fixed.*

Proof. First note that the probability density function between two points X_1 and X_2 from two different slices is

$$p(x_1, x_2) = \frac{n}{n-m}(\frac{t-1}{t} - \delta_n(x_1, x_2) + \frac{1}{t}\gamma_m(x_1, x_2)). \tag{1.5.2}$$

where

$$\delta_n(x_1, x_2) = \begin{cases} 1, & \lceil nx_1 \rceil = \lceil nx_2 \rceil, \\ 0, & otherwise; \end{cases}$$

and

$$\gamma_m(x_1, x_2) = \begin{cases} 1, & \lceil mx_1 \rceil = \lceil mx_2 \rceil, \\ 0, & otherwise. \end{cases}$$

For the covariance between X_1 and X_2, we have

$$\begin{aligned} cov(X_1, X_2) &= EX_1X_2 - EX_1EX_2 \\ &= \frac{n}{n-m}\int_0^1\int_0^1 x_1x_2(\frac{t-1}{t} - \delta_n(x_1, x_2) + \frac{1}{t}\gamma_m(x_1, x_2))dx_1dx_2 - EX_1EX_2 \\ &= \frac{n}{n-m}\int_0^1\int_0^1 -x_1x_2\delta_n(x_1, x_2) + \frac{1}{t}x_1x_2\gamma_m(x_1, x_2)dx_1dx_2. \end{aligned} \tag{1.5.3}$$

Now we look at $\int_0^1 \int_0^1 x_1 x_2 \delta_n(x_1, x_2) dx_1 dx_2$ and $\int_0^1 \int_0^1 x_1 x_2 \gamma_m(x_1, x_2) dx_1 dx_2$ separately. For $\int_0^1 \int_0^1 x_1 x_2 \delta_n(x_1, x_2) dx_1 dx_2$, we have

$$\begin{aligned}
& \int_0^1 \int_0^1 x_1 x_2 \delta_n(x_1, x_2) dx_1 dx_2 \\
= & \sum_{i=1}^{n} \sum_{j=1}^{n} \int_{\frac{i-1}{n}}^{\frac{i}{n}} \int_{\frac{j-1}{n}}^{\frac{j}{n}} x_1 x_2 \delta_n(x_1, x_2) dx_1 dx_2 \\
= & \sum_{i=1}^{n} \int_{\frac{i-1}{n}}^{\frac{i}{n}} x_1 dx_1 \int_{\frac{i-1}{n}}^{\frac{i}{n}} x_2 dx_2 \\
= & \sum_{i=1}^{n} \frac{1}{2} \frac{i^2 - (i-1)^2}{n^2} \frac{1}{2} \frac{i^2 - (i-1)^2}{n^2} \\
= & \frac{1}{4n^4} \sum_{i=1}^{n} (2i-1)^2 \\
= & \frac{1}{4n^4} (\frac{4}{3} n^3 + 2n^2 + \frac{2}{3} n - 4 \frac{n(n+1)}{2} + n) \\
= & \frac{1}{3n} - \frac{1}{12n^3}. \qquad (1.5.4)
\end{aligned}$$

Similarly for the second term we have

$$\int_0^1 \int_0^1 x_1 x_2 \gamma_m(x_1, x_2) dx_1 dx_2 = \frac{1}{3m} - \frac{1}{12m^3}. \qquad (1.5.5)$$

So we have

$$\begin{aligned}
& cov(X_1, X_2) \\
= & \frac{n}{n-m} (\frac{1}{3mt} - \frac{1}{12m^3 t} - \frac{1}{3n} + \frac{1}{12n^3}) \\
= & \frac{n}{n-m} (\frac{1}{12n^2} - \frac{1}{12m^2}) \\
= & -\frac{t+1}{12m^3 t^2} < 0. \qquad (1.5.6)
\end{aligned}$$

It is easy to see from (1.5.6) that $cov(X_1, X_2)$ is an increasing function of t. □

From (1.5.6), the following is observed:

(i). The covariance is non-positive.

(ii). With m fixed, as $t \to \infty$, $cov(X_1, X_2) \to 0$. This convergence is monotonic and of order $O(t^{-1})$.

(iii). With t fixed, as $m \to \infty$, $cov(X_1, X_2) \to 0$. This convergence is monotonic and of order $O(m^{-3})$.

(iv). With n fixed, as m decreases, $cov(X_1, X_2)$ decreases, or equivalently, the magnitude of $cov(X_1, X_2)$ increases.

(v). It can be shown that the covariance between two points in an ordinary LHD on $(0, 1]$ is $-\frac{n+1}{12n^2}$. So when $m = 1, t = n$, the covariance formula for an SLHD becomes the counterpart for an ordinary LHD.

The following corollary says that the magnitude of the covariance between points in the same slice of an SLHD on $(0, 1]$ is higher than that between points in different slices.

Corollary 1.5.1. *For an SLHD on* $(0, 1]$*, let t denote the number of slices and m the number of runs in each slice. For any two points* X_1 *and* X_2 *from the same slice and any two points* X_3 *and* X_4 *from two different slices, we have*

$$cov(X_1, X_2) < cov(X_3, X_4) < 0. \tag{1.5.7}$$

Proof. From (1.5.6), we have $cov(X_3, X_4) = -\frac{t+1}{12m^3t^2}$. Since each slice is an ordinary LHD, we have $cov(X_1, X_2) = -\frac{m+1}{12m^2} = -\frac{n^2+nt}{12m^3t^2}$. It is then obvious that (1.5.7) holds. □

We now explore the covariance structure between different rows of a DSLHD.

Proposition 1.5.2. *For a DSLHD(m, s, t, 1) on* $(0, 1]$*, the covariance between any two points from two different rows is non-positive and increases as t or s increases with m fixed.*

Proof. Using the notation in Section 1.2, we can consider, without loss of generality, the covariance between D_{11} and D_{21}. Since the DSLHD is on $(0, 1]$, D_{11} and D_{21} are two

scalars. For ease of notation, we use X_1 and X_2 to denote D_{11} and D_{21} respectively. From Lemma 1.3.1 (iv), the joint density function of X_1 and X_2 is

$$p(x_1, x_2) = 1 + \frac{1}{s-1}\delta_{mt}(x_1, x_2) - \frac{s}{s-1}\delta_n(x_1, x_2), \tag{1.5.8}$$

where $\delta_{mt}(x_1, x_2)$ and $\delta_n(x_1, x_2)$ are similarly defined as in Proposition 1.5.1. The derivation of this joint density function is in the appendix. For the covariance between X_1 and X_2, we have

$$\begin{aligned} & cov(X_1, X_2) = EX_1X_2 - EX_1EX_2 \\ = \quad & \frac{1}{s-1}\int_0^1\int_0^1 x_1x_2\delta_{mt}(x_1, x_2)dx_1dx_2 - \frac{s}{s-1}\int_0^1\int_0^1 x_1x_2\delta_n(x_1, x_2)dx_1dx_2 \end{aligned} \tag{1.5.9}$$

Using similar calculations as in Proposition 1.5.1, we have

$$cov(X_1, X_2) = \frac{1}{12(s-1)(mt)^3}(\frac{1}{s^2} - 1) < 0. \tag{1.5.10}$$

The conclusion is thus obvious from (1.5.10). □

We have the following observations from (1.5.10).

(i). The covariance is always non-positive.

(ii). For fixed m and t, as $s \to \infty$, $cov(X_1, X_2) \to 0$. The convergence is monotonic and of order $O(s^{-1})$.

(iii). For fixed m and s, as $t \to \infty$, $cov(X_1, X_2) \to 0$. The convergence is monotonic and of order $O(t^{-3})$.

(iv). For fixed s and t, as $m \to \infty$, $cov(X_1, X_2) \to 0$. The convergence is monotonic and of order $O(m^{-3})$.

(v). For $t = 1$, (1.5.10) becomes (1.5.6).

(vi). For fixed n and m, as s increases, (1.5.10) decreases. The explanation is as follows.

When n and m are fixed, st is also fixed. Let $a = st$. From (1.5.10), we have

$$\begin{aligned} & \frac{1}{12m^3(s-1)t^3}(\frac{1}{s^2}-1) = \frac{1}{12m^3}(\frac{1}{s^2t^3(s-1)} - \frac{1}{(s-1)t^3}) \\ = & \frac{1}{12m^3}(\frac{1}{a^3-a^2t} - \frac{1}{at^2-t^3}) = \frac{1}{12m^3}(-\frac{1}{at^2} - \frac{1}{a^2t}). \end{aligned} \tag{1.5.11}$$

The conclusion is obvious from the above formula.

1.5.2 A numerical study

In this subsection, we run some numerical experiments to compare the DSLH scheme and the three SLH schemes. The theoretical results in Section 1.5.1 will be utilized to help better understand the implications of the numerical results.

Example 1.3. This example uses a five-dimensional function ([16])

$$f(\mathbf{x}) = log(x_1x_2x_3x_4x_5), \tag{1.5.12}$$

where $X = (x_1, x_2, x_3, x_4, x_5)$ is uniformly distributed on $[0, 1]^5$. Suppose we want to run this model in a batch sequential way with a fixed batch size. The objective is to estimate the mean of the output given the distribution of the inputs. For illustration purpose, suppose we would like to run the experiments in four batches with a fixed batch size of m. We are interested in the statistical properties of sample average of the output after running one batch, two batches and four batches respectively. Let μ be the true mean of the function. Let $\hat{\mu}_{11}$, $\hat{\mu}_{12}$, $\hat{\mu}_{21}$ and $\hat{\mu}_{22}$ be the sample average of the output for the first batch, the second batch, the third batch and the fourth batch respectively. Let $\lambda_{11} = \lambda_{12} = \lambda_{21} = \lambda_{22} = 0.25$. Therefore, we are interested in the statistical properties of $\hat{\mu}_{11}$ and the following two quantities:

$$\hat{\eta}_1 = \lambda_{11}\hat{\mu}_{11} + \lambda_{12}\hat{\mu}_{12},$$

and

$$\hat{\eta} = \lambda_{11}\hat{\mu}_{11} + \lambda_{12}\hat{\mu}_{12} + \lambda_{21}\hat{\mu}_{21} + \lambda_{22}\hat{\mu}_{22}.$$

Table 1.5.1: RMSEs of $\hat{\mu}_{11}$ in Example 1.3

	DSLH	SLH1	SLH2	SLH3
$m = 5$	0.4671	0.4756	0.4597	0.8002
$m = 10$	0.2310	0.2306	0.2337	0.5215
$m = 20$	0.1154	0.1134	0.1192	0.3646
$m = 40$	0.0576	0.0584	0.0608	0.2527

We compare DSLH, SLH1, SLH2, and SLH3 in Definition 2 for this purpose. For DSLH, we generate a DSLHD(m,2,2,5) and assign D_{11} as the first batch, D_{12} as the second batch, D_{21} as the third batch, D_{22} as the fourth batch. For SLH1, we generate an SLHD of four slices, each has a size of m and then treat the four slices as four batches. For SLH2, we independently generate two SLHDs, each of which has two slices with slice size of m. We then treat the first slice in the first SLHD as the first batch, the second slice in the first SLHD as the second batch, the first slice in the second SLHD as the third batch and the second slice in the second SLHD as the fourth batch. For SLH3, we generate an SLHD of two slices, each of which has a size of $2m$. We then randomly split the first slice into two subsets of m runs and use the first subset as the first batch and the second subset as the second batch. Similarly we randomly split the second slice into two subsets of m runs and use the first subset as the third batch and the second subset as the fourth batch.

For each scheme, we computed $\hat{\mu}_{11}$, $\hat{\eta}_1$ and $\hat{\eta}$ 2000 times for $m = 5, 10, 20, 40$. Tables 1.5.1, 1.5.2 and 1.5.3 present the RMSE of $\hat{\mu}_{11}$, $\hat{\eta}_1$ and $\hat{\eta}$ respectively. These tables clearly show that, for every value of m, the DSLH scheme achieves the most variance reduction of all three estimates. However, the three SLH schemes all have their drawbacks. Specifically, $\hat{\mu}_{11}$ under the SLH3 scheme has a significantly larger variance than that under the other three schemes; $\hat{\eta}_1$ under the SLH1 scheme has a significantly larger variance than the other three schemes; $\hat{\eta}$ under the SLH2 scheme has a significantly larger variance than the other three schemes.

We now provide an example of a set of different computer models and compare the four

Table 1.5.2: RMSEs of $\hat{\eta}_1$ in Example 1.3

	DSLH	SLH1	SLH2	SLH3
$m = 5$	0.1177	0.1403	0.1137	0.1183
$m = 10$	0.0579	0.0720	0.0586	0.0585
$m = 20$	0.0284	0.0355	0.0298	0.0299
$m = 40$	0.0150	0.0177	0.0147	0.0145

Table 1.5.3: RMSEs of $\hat{\eta}$ in Example 1.3

	DSLH	SLH1	SLH2	SLH3
$m = 5$	0.1140	0.1143	0.1640	0.1203
$m = 10$	0.0570	0.0579	0.0811	0.0574
$m = 20$	0.0296	0.0288	0.0412	0.0295
$m = 40$	0.0146	0.0145	0.0210	0.0149

schemes.

Example 1.4. This example uses the following four functions:

$$\begin{aligned} f_{11}(\mathbf{x}) &= log(\frac{1}{\sqrt{x_1}} + \frac{1}{\sqrt{x_2}}), \\ f_{12}(\mathbf{x}) &= log(\frac{0.98}{\sqrt{x_1}} + \frac{0.95}{\sqrt{x_2}}), \\ f_{21}(\mathbf{x}) &= log(\frac{1.02}{\sqrt{x_1}} + \frac{1.02}{\sqrt{x_2}}), \\ f_{22}(\mathbf{x}) &= log(\frac{1}{\sqrt{x_1}} + \frac{1.03}{\sqrt{x_2}}). \end{aligned}$$

Suppose the four functions can be partitioned into two groups such that f_{11} and f_{12} are two variants of the first computer model and f_{21} and f_{22} are two variants of the second computer model. We are interested in the mean of each variant as well as the following three quantities:

$$\eta = \frac{1}{4}\mu_{11} + \frac{1}{4}\mu_{12} + \frac{1}{4}\mu_{21} + \frac{1}{4}\mu_{22},$$

$$\eta_1 = \frac{1}{2}\mu_{11} + \frac{1}{2}\mu_{12},$$

$$\eta_2 = \frac{1}{2}\mu_{21} + \frac{1}{2}\mu_{22}.$$

We compare schemes SLH1, SLH2, SLH3 and DSLH for estimating these parameters. The setup is the same as in Example 1.3. The RMSEs of $\hat{\mu}_{11}$, $\hat{\eta}_1$, $\hat{\eta}_2$ and $\hat{\eta}$ are shown in Tables 1.5.4-1.5.7 respectively. The key takeaway is that the DSLH scheme works well for estimation of all the four parameters, while each SLH scheme has its disadvantage. Specifically, SLH1 does not estimate η_1 and η_2 as efficiently, SLH2 does not estimate η as efficiently and SLH3 does not estimate η as efficiently.

We now provide some comments on Tables 1.5.4-1.5.7.

(i). From Table 1.5.7, we observe that DSLH, SLH1 and SLH3 perform similarly for the estimation of η. This is because the four functions used in this example are similar to each other. Moreover, both SLHD and DSLHD are an LHD when viewed as an overall design. The three schemes all work better than SLH2 because the two design sets for estimating η_1 and η_2 under the SLH2 scheme are independent, while under the other three schemes a point in the design set for estimating η_1 is negatively correlated with a point in that for estimating η_2.

(ii). From Table 1.5.5, it is observed that the DSLH scheme outperforms the SLH1 scheme for the estimation of η_1. Heuristically, this is because the two slices used to estimate η_1 in the DSLH scheme form an LHD, while the two slices in the SLH1 scheme do not have this property. A more rigorous explanation is as follows. Let $\mathbf{D}_1^{SLH1}$, $\mathbf{D}_2^{SLH1}$ and $\mathbf{D}_1^{DSLH}$, $\mathbf{D}_2^{DSLH}$ denote the two slices used to estimate η_1 in the SLH scheme and the DSLH scheme respectively. Denote the data points in $\mathbf{D}_1^{SLH1}$, $\mathbf{D}_2^{SLH1}$, $\mathbf{D}_1^{DSLH}$, and $\mathbf{D}_2^{DSLH}$ as $\mathbf{x}_{11}^{SLH1}, \ldots, \mathbf{x}_{1m}^{SLH1}$; $\mathbf{x}_{21}^{SLH1}, \ldots, \mathbf{x}_{2m}^{SLH1}$; $\mathbf{x}_{11}^{DSLH}, \ldots, \mathbf{x}_{1m}^{DSLH}$; and $\mathbf{x}_{21}^{DSLH}, \ldots, \mathbf{x}_{2m}^{DSLH}$ respectively. We have, for the variance of $\hat{\eta}_1$ under the SLH1

scheme,

$$\begin{aligned} & var_{SLH1}(\hat{\eta}_1) = var_{SLH1}(\hat{\mu}_{11}) + var_{SLH1}(\hat{\mu}_{12}) + cov_{SLH1}(\hat{\mu}_{11}, \hat{\mu}_{12}) \\ = \; & var_{SLH1}(\hat{\mu}_1) + var_{SLH1}(\hat{\mu}_2) + cov_{SLH1}(f_{11}(\mathbf{x}_{11}^{SLH1}), f_{12}(\mathbf{x}_{21}^{SLH1})). \end{aligned} \quad (1.5.13)$$

Similarly, for the variance of $\hat{\eta}_1$ under the DSLH scheme, we have

$$\begin{aligned} & var_{DSLH}(\hat{\eta}_1) \\ = \; & var_{DSLH}(\hat{\mu}_{11}) + var_{DSLH}(\hat{\mu}_{12}) + cov_{DSLH}(\hat{\mu}_{11}, \hat{\mu}_{12}) \\ = \; & var_{DSLH}(\hat{\mu}_{11}) + var_{DSLH}(\hat{\mu}_{12}) + cov_{DSLH}(f_{11}(\mathbf{x}_{11}^{DSLH}), f_{12}(\mathbf{x}_{21}^{DSLH})) \end{aligned} \quad (1.5.14)$$

From Lemma 1.3.2, a slice in an SLHD is statistically equivalent to the counterpart in a DSLHD. Hence we have $var_{SLH1}(\hat{\mu}_{11}) = var_{DSLH}(\hat{\mu}_{11})$ and $var_{SLH1}(\hat{\mu}_{12}) = var_{DSLH}(\hat{\mu}_{12})$. From Lemma 1.3.3, a row in a DSLHD is statistically equivalent to an SLHD of the same size. This fact together with Proposition 1.4.1 says that $cov_{DSLH}(\mathbf{x}_{11}^{DSLH}, \mathbf{x}_{21}^{DSLH}) < cov_{SLH1}(\mathbf{x}_{11}^{SLH1}, \mathbf{x}_{21}^{SLH1})$. This follows because $\mathbf{x}_{11}^{DSLH}, \mathbf{x}_{21}^{DSLH}$ are from two different slices of an SLHD that has two slices, each of which has m runs, while $\mathbf{x}_{11}^{SLH1}, \mathbf{x}_{21}^{SLH1}$ come from two different slices of an SLHD that has four slices, each of which has m runs. Since f_{11} and f_{12} are jointly decreasing in both arguments of $\mathbf{x}$, we conjecture that

$$cov_{DSLH}(f_{11}(\mathbf{x}_{11}^{DSLH}), f_{12}(\mathbf{x}_{21}^{DSLH})) < cov_{SLH1}(f_{11}(\mathbf{x}_{11}^{SLH1}), f_{12}(\mathbf{x}_{21}^{SLH1})) < 0. \quad (1.5.15)$$

This gives a heuristic explanation on the inequality $var_{DSLH}(\hat{\eta}_1) < var_{SLH1}(\hat{\eta}_1)$. Similarly we can explain why DSLH outperforms SLH1 for estimating η_2.

(iii). DSLH outperforms SLH3 for estimating μ_{11} because SLH3 randomly split an ordinary LHD to different subsets and the design set for estimating μ_{11} is not guaranteed to be an LHD.

Note from the theoretical results in Section 1.5.1, we can rigorously prove that, the DSLH scheme is better than the SLH1 scheme if the subset of functions of interest are

Table 1.5.4: RMSEs of $\hat{\mu}_{11}$ in Example 1.4

	DSLH	SLH1	SLH2	SLH3
$m = 5$	0.1165	0.1149	0.1161	0.1581
$m = 10$	0.0629	0.0577	0.0620	0.1055
$m = 20$	0.0326	0.0338	0.0329	0.0710
$m = 40$	0.0184	0.0184	0.0178	0.0495

Table 1.5.5: RMSEs of $\hat{\eta}_1$ in Example 1.4

	DSLH	SLH1	SLH2	SLH3
$m = 5$	0.0320	0.0367	0.0320	0.0300
$m = 10$	0.0171	0.0185	0.0166	0.0173
$m = 20$	0.0092	0.0104	0.0091	0.0089
$m = 40$	0.0050	0.0060	0.0051	0.0049

Table 1.5.6: RMSEs of $\hat{\eta}_2$ in Example 1.4

	DSLH	SLH1	SLH2	SLH3
$m = 5$	0.0287	0.0359	0.0318	0.0312
$m = 10$	0.0158	0.0197	0.0165	0.0169
$m = 20$	0.0090	0.0102	0.0094	0.0092
$m = 40$	0.0051	0.0057	0.0053	0.0052

Table 1.5.7: RMSEs of $\hat{\eta}$ in Example 1.4

	DSLH	SLH1	SLH2	SLH3
$m = 5$	0.0337	0.0344	0.0454	0.0331
$m = 10$	0.0181	0.0186	0.0235	0.0188
$m = 20$	0.0103	0.0103	0.0132	0.0105
$m = 40$	0.0059	0.0059	0.0073	0.0060

Table 1.5.8: RMSEs of $\hat{\mu}_{11}$ in Example 1.5

	DSLH	SLH1	SLH2	SLH3
$m = 5$	0.1175	0.1125	0.1103	0.1568
$m = 10$	0.0624	0.0601	0.0617	0.1082
$m = 20$	0.0324	0.0327	0.0326	0.0708
$m = 40$	0.0180	0.0187	0.0191	0.0501

Table 1.5.9: RMSEs of $\hat{\eta}_1$ in Example 1.5

	DSLH	SLH1	SLH2	SLH3
$m = 5$	0.0314	0.0347	0.0296	0.0298
$m = 10$	0.0169	0.0188	0.0168	0.0170
$m = 20$	0.0090	0.0106	0.0088	0.0094
$m = 40$	0.0051	0.0059	0.0052	0.0052

all linear and jointly increasing or decreasing in each argument of $\mathbf{x}$. More generally, we conjecture that the advantage of the DSLH scheme over the SLH1 scheme will be more significant if the subset of functions are additive and jointly increasing or decreasing in each argument of $\mathbf{x}$. We use the next example to illustrate this point.

Example 1.5. The four functions used in this example are as follows:

$$\begin{aligned} f_{11}(\mathbf{x}) &= log(\frac{1}{\sqrt{x_1}} + \frac{1}{\sqrt{x_2}}), \\ f_{12}(\mathbf{x}) &= log(\frac{0.98}{\sqrt{x_1}} + \frac{0.95}{\sqrt{x_2}}), \\ f_{21}(\mathbf{x}) &= -x_1 - x_2, \\ f_{22}(\mathbf{x}) &= -x_1^2 - x_2^2. \end{aligned} \tag{1.5.16}$$

Suppose f_{11} and f_{12} are two variants of a computer model and f_{21} and f_{22} are two variants of another computer model. The setup and notations are exactly the same as in Example 1.4. We show the numerical results in Tables 1.5.8-1.5.11.

The conclusions are the same as in Example 1.4. That is, the DSLH scheme estimates all the four parameters simultaneously well, while each of the SLH schemes has its own drawback. The extra takeaway here is that the advantage of the DSLH scheme over the

Table 1.5.10: RMSEs of $\hat{\eta}_2$ in Example 1.5

	DSLH	SLH1	SLH2	SLH3
$m = 5$	0.0084	0.0116	0.0084	0.0109
$m = 10$	0.0030	0.0042	0.0029	0.0065
$m = 20$	0.0010	0.0015	0.0010	0.0043
$m = 40$	3.78e-004	5.32e-004	3.60e-004	0.0030

Table 1.5.11: RMSEs of $\hat{\eta}$ in Example 1.5

	DSLH	SLH1	SLH2	SLH3
$m = 5$	0.0314	0.0338	0.0308	0.0306
$m = 10$	0.0169	0.0185	0.0170	0.0180
$m = 20$	0.0090	0.0105	0.0089	0.0102
$m = 40$	0.0051	0.0059	0.0052	0.0059

SLH1 scheme is more pronounced for the estimation of η_2 than for η_1. This is because f_{21} is a linear function and f_{22} is an additive function.

1.6 Multi-layer sliced Latin hypercube design (MLSLHD)

In this section, we generalize the slicing structure to multiple layers and propose a more general class of designs called multi-layer sliced Latin hypercube design (MLSLHD). SLHD and DSLHD can be viewed as an MLSLHD with one and two layers respectively. We propose a recursive strategy to construct MLSLHD with an arbitrary number of layers. Without loss of generality, assume the design space is $[0, 1]^d$. Suppose the number of runs n can be written as $n = m \prod_{k=1}^{r} s_k$, where $s_1, \ldots, s_r$, m and r are positive integers. An r-layer MLSLHD $\mathbf{D}$ with d factors has the following form: In the first layer, $\mathbf{D}$ consists of $\prod_{k=1}^{r} s_k$ small LHDs, each of which has m runs. In the second layer, there are $\prod_{k=2}^{r} s_k$ LHDs, each of which is of ms_1 levels and consists of s_1 LHDs from the first layer. Similarly, in the kth layer for $k = 3, \ldots, r$, there are $\prod_{j=k}^{r} s_j$ LHDs, each of which is of $m \prod_{j=1}^{k-1} s_j$ levels and consists of s_k LHDs from the $(k-1)$th layer. The s_r LHDs in the rth layer constitute the whole design, which is also an LHD. We denote such a design by MLSLHD$(s_1, \ldots, s_r;\ m;\ d)$.

Here $s_1, \ldots, s_r$ represent the layer structure, m is the number of runs in each LHD in the first layer and d is the number of factors. The whole design can be partitioned into $\prod_{j=k}^{r} s_j$ LHDs in the kth layer for $k = 1, \ldots, r$. An MLSLHD$(s_1, s_2, \ldots, s_r;\ m;\ d)$ reduces to an SLHD and a DSLHD for $r = 1$ and $r = 2$ respectively. Moreover, an ordinary LHD without any slicing structure can be viewed as an r-layer MLSLHD with $r = 0$. A tree-structure example of a three-layer MLSLHD of with $s_1 = 2$, $s_2 = 3$ and $s_3 = 2$ is shown in Figure 1.6.1. A three-layer MLSLHD example is shown in Figure 1.6.2. All markers in the

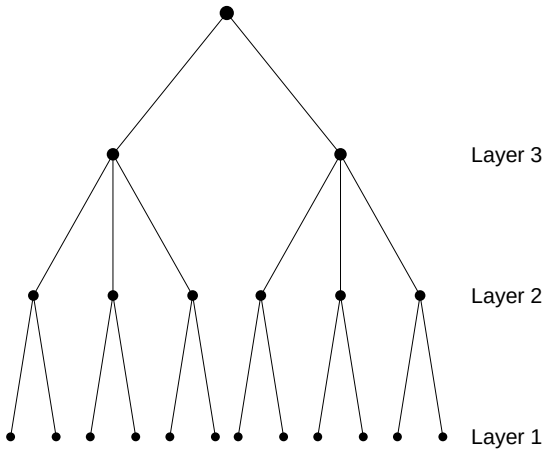

Figure 1.6.1: Tree diagram for a three-layer MLSLHD with $s_1 = 2,\ s_2 = 3$ and $s_3 = 2$

figure constitute an MLSLHD$(2, 2, 2;\ 3;\ 2)$, which is a larger LHD. In the first layer, the MLSLHD$(2, 2, 2;\ 3;\ 2)$ is partitioned into eight small LHDs of three runs, denoted by small red circles, large red circles, small red stars, large red stars, small blue circles, large blue circles, small blue stars, and large blue stars, respectively. In the second layer, there are four LHDs, denoted by red circles, red stars, blue circles, and blue stars, respectively. In the third layer, there are two LHDs, denoted by red markers and blue markers, respectively.

Before presenting the method for constructing MLSLHD$(s_1, \ldots, s_r;\ m;\ d)$, we define the r-layer sliced permutation vector PM$(s_1, \ldots, s_r;\ m)$ corresponding to MLSLHD$(s_1, \ldots, s_r;\ m; d)$. When r is one, the entries $p_1, \ldots, p_n$ of PM$(s_1;\ m)$ is a permutation on $\mathbf{Z}_n$. Moreover,

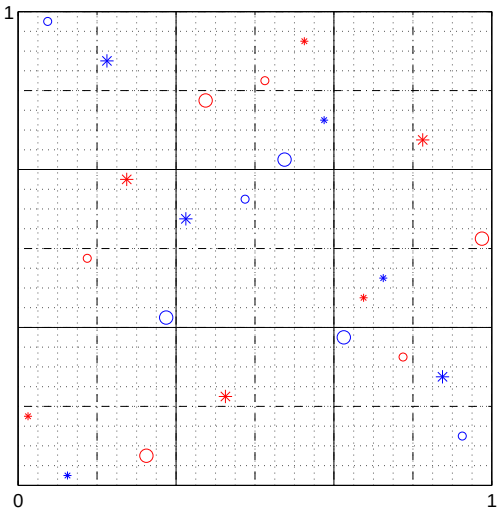

Figure 1.6.2: A three-layer MLSLHD example

for $i = 1, \ldots, s_1$, $\{\lceil p_{(i-1)m+1}/s_1 \rceil, \ldots, \lceil p_{(i-1)m+m}/s_1 \rceil\}$ is a permutation on $\mathbf{Z}_m = \{1, \ldots, m\}$. These s_1 sets form a segmentation of PM$(s_1;\ m)$ and each of them is a permutation vector for an ordinary LHD, corresponding to a slice of the one layer MSLHD with PM$(s_1;\ m)$ as the permutation vector. For $r > 1$, a sliced permutation PM$(s_1, \ldots, s_r;\ m)$ with entries $p_1, \ldots, p_n$ is a permutation on $\mathbf{Z}_n$, where for $i = 1, \ldots, s_r$, $\{\lceil p_{(i-1)w+1}/s_r \rceil, \ldots, \lceil p_{(i-1)w+w}/s_r \rceil\}$ is a PM$(s_1, \ldots, s_{r-1};\ m)$. Here $w = m \prod_{k=1}^{r-1} s_k$.

Let's look at the three-layer example in Figure 1.6.1. A PM$(2, 3, 2;\ m)$ can be partitioned into two segments $(\mathbf{b}_1', \mathbf{b}_2')$ corresponding to two LHDs in the third layer, where $\lceil \mathbf{b}_1/2 \rceil$ and $\lceil \mathbf{b}_2/2 \rceil$ have to be PM$(2, 3;\ m)$. Therefore we can first construct two PM$(3, 2;\ m)$'s. Similarly, to construct a PM$(2, 3;\ m)$, we can construct three PM$(2;\ m)$'s, and each PM$(2;\ m)$ needs two permutations on $\mathbf{Z}_m$. We now make this statement more general in the following description.

Step 1: Construct the $w \times s_r$ matrix $\boldsymbol{H} = (h_{ij})$, whose ith row is $((i-1)s_r+1, \ldots, (i-1)s_r+s_r)$.

Step 2: Construct the $w \times s_r$ matrix $\boldsymbol{C} = (c_{ij})$, whose ith row is a uniform permutation on $(h_{i1}, \ldots, h_{is_r})$. Here the permutations are carried out independently from one row to

another.

Step 3: For $i = 1, \ldots, s_r$,

If $r = 1$,

Generate $(p_1, \ldots, p_w)'$ to be a permutation on $\mathbf{Z}_w$.

Else

Suppose that a PM$(s_1, \ldots, s_{r-1};\ m)$ has been generated. Denote it by $(p_1, \ldots, p_w)'$.

Construct a vector $\mathbf{b}_i = (b_{1i}, \ldots, b_{wi})'$ that is a permutation on $(c_{1i}, \ldots, c_{wi})'$ by

$$b_{ji} = c_{p_j i} \text{ for } j = 1, \ldots, w.$$

End If

End i loop.

All the implementations in Step 3 are carried out independently to each other.

Step 4: A PM$(s_1, \ldots, s_r;\ m)$ is given by $(\mathbf{b}_1', \ldots, \mathbf{b}_{s_r}')'$.

We now use two examples to illustrate the construction procedure.

Example 1.6 In this example, we generate a two-layer SPV$(2, 2;\ 3)$. Suppose the matrix $\boldsymbol{C}$ (in transpose) in Step 2 has been generated as follows:

$$\boldsymbol{C}^T = \begin{pmatrix} 1 & 4 & 6 & 7 & 9 & 12 \\ 2 & 3 & 5 & 8 & 10 & 11 \end{pmatrix}.$$

For Step 3, since $s_2 = 2$, we need to generate two independent SPV$(2;\ 3)$s first. Using the algorithm in [47], suppose we have generated two SPV$(2;\ 3)$s $(1, 3, 6, 4, 2, 5)'$ and $(1, 5, 4, 6, 2, 3)'$. Note both SPV$(2;\ 3)$s are concatenation of two vectors $\boldsymbol{v}_1$ and $\boldsymbol{v}_2$ such that $\lceil \boldsymbol{v}_k/2 \rceil$ is a permutation vector of an ordinary LHD of three levels. Here $k = 1, 2$. Given

the generated SPV(2; 3)s, $\mathbf{b}_1$ and $\mathbf{b}_2$ in Step 3 are $(1, 6, 12, 7, 4, 9)'$ and $(2, 10, 8, 11, 3, 5)'$ separately. Thus the desired two-layer SPV(2, 2; 3) is $(1, 6, 12, 7, 4, 9, 2, 10, 8, 11, 3, 5)'$.

Example 1.7 We generate an SPV(2, 2, 2; 3) in this example. Suppose the matrix $\boldsymbol{C}$ (in transpose) in Step 2 has been generated as follows:

$$\boldsymbol{C}^T = \begin{pmatrix} 1 & 4 & 6 & 7 & 10 & 11 & 13 & 15 & 18 & 20 & 21 & 24 \\ 2 & 3 & 5 & 8 & 9 & 12 & 14 & 16 & 17 & 19 & 22 & 23 \end{pmatrix}.$$

Since $s_3 = 2$, we need to generate two independent SPV(2, 2; 3). Using the same strategy as in Example 1.6, suppose we have generated two SPV(2, 2; 3)s $(7, 2, 10, 12, 4, 5, 1, 9, 8, 11, 6, 3)'$ and $(6, 1, 12, 9, 4, 7, 8, 2, 10, 11, 3, 5)'$. $\mathbf{b}_1$ and $\mathbf{b}_2$ in Step 3 are thus $(13, 4, 20, 24, 7, 10, 1, 18, 15, 21, 11, 6)'$ and $(12, 2, 23, 17, 8, 14, 16, 3, 19, 22, 5, 9)'$ respectively. Hence the desired SPV(2, 2, 2; 3) is $(13, 4, 20, 24, 7, 10, 1, 18, 15, 21, 11, 6, 12, 2, 23, 17, 8, 14, 16, 3, 19, 22, 5, 9)'$. Note that the generated $SPV(2, 2, 2;\ 3)$ can be partitioned into eight vectors: $\mathbf{v}_{111} = (13, 4, 20)'$, $\mathbf{v}_{112} = (24, 7, 10)'$, $\mathbf{v}_{121} = (1, 18, 15)'$, $\mathbf{v}_{122} = (21, 11, 6)'$, $\mathbf{v}_{211} = (12, 2, 23)'$, $\mathbf{v}_{212} = (17, 8, 14)'$, $\mathbf{v}_{221} = (16, 3, 19)'$ and $\mathbf{v}_{222} = (22, 5, 9)'$. At the first layer, $\lceil \mathbf{v}_{ijk}/8 \rceil$ is a permutation on $\boldsymbol{Z}_3$, $i, j, k = 1, 2$. At the second layer, let $\mathbf{v}_{11} = (\mathbf{v}'_{111}, \mathbf{v}'_{112})' = (13, 4, 20, 24, 7, 10)'$, $\mathbf{v}_{12} = (\mathbf{v}'_{121}, \mathbf{v}'_{122})' = (1, 18, 15, 21, 11, 6)'$, $\mathbf{v}_{21} = (\mathbf{v}'_{211}, \mathbf{v}'_{212})' = (12, 2, 23, 17, 8, 14)'$ and $\mathbf{v}_{22} = (\mathbf{v}'_{221}, \mathbf{v}'_{222})' = (16, 3, 19, 22, 5, 9)'$. $\lceil \mathbf{v}_{ij}/4 \rceil$ is a permutation on $\boldsymbol{Z}_6$, $i, j = 1, 2$. At the third layer, let $\mathbf{v}_1 = (\mathbf{v}'_{11}, \mathbf{v}'_{12})' = (13, 4, 20, 24, 7, 10, 1, 18, 15, 21, 11, 6)$ and $\mathbf{v}_2 = (\mathbf{v}'_{21}, \mathbf{v}'_{22})' = (12, 2, 23, 17, 8, 14, 16, 3, 19, 22, 5, 9)$. $\lceil \mathbf{v}_i/2 \rceil$ is a permutation on $\boldsymbol{Z}_{12}$.

Based on an r-layer sliced permutation vector PM($s_1, \ldots, s_r$; m), an r-layer MLSLHD($s_1, \ldots, s_r$; m; d) can be generated as follows.

Step 1: Construct the $n \times d$ matrix $\mathbf{A} = (a_{ij})$, whose columns are generated independently from d PM($s_1, \ldots, s_r$; m)'s.

Step 2: Construct the $n \times d$ matrix $\mathbf{D}$, the (i, j)th entry of which is

$$d_{ij} = \frac{a_{ij} - u_{ij}}{n}, \quad \text{for } i = 1, \ldots, n,\ j = 1, \ldots, d,$$

where the u_{ij} are i.i.d. $U[0,1)$ random variables, and the a_{ij} and the u_{ij} are mutually independent. $\mathbf{D}$ is an r-layer MLSLHD$(s_1, \ldots, s_r;\ m;\ d)$.

1.7 Concluding remarks

We have proposed a new class of designs, called doubly sliced Latin hypercube design, for running computer experiments. An SLHD ([47]) is a special case of a doubly sliced Latin hypercube design. Potential applications of DSLHDs, other than computer experiments, include cross-validation, stochastic optimization and selection of the tuning parameter in smoothing splines regression or other similar nonparametric regression methods. For details, see [15], [14].

In Section 1.5, we compared DSLHDs and SLHDs using a few numerical examples and some heuristic arguments. Asymptotically, the DSLH scheme and the SLH scheme are similar. The finite sample comparison would be more interesting and warrants further work.

Another interesting research direction is to obtain a central limit theorem for doubly sliced Latin hypercube sampling. Central limit theorems for Latin hypercube sampling have been given in simpler situations ([60]; [43]; [35]). The central limit theorem for doubly sliced Latin hypercube sampling is much more technically involved because of the complicated covariance structure among the sampled points.

In Section 1.6, we further extend the sliced structure to MLSLHD of arbitrary number of layers. The application of MLSLHD to real problem warrants further work. The sliced structure should depend on the specific application. Higher number of layers means more flexibility in allocation of design points but more constraints on the size of design.

In the next chapter, we propose some sequential designs based on DSLHD.

1.8 Appendix

1.8.1 Proof of Lemma 1.3.1

Proof. First note that the dimensions of $\mathbf{H}$ are exchangeable rowwise, columnwise and elementwise. To prove (i), it suffices to consider h_{111} because of the exchangeability. By symmetry, $Pr(h_{111} = u)$ takes the same value for all $u \in \mathbf{Z}_n$. So we have $Pr(h_{111} = u) = 1/n$. Thus, (1.3.7) holds.

To prove (ii), it suffices to consider h_{111} and h_{211} because of the exchangeability of H. Since the set $h(:, 1, 1)$ is an LHD of m levels, we have for u, v with $\lceil u/st \rceil = \lceil v/st \rceil$, $Pr(h_{111} = u, h_{211} = v) = 0$. Because there are $n(n-st)$ (u, v)s that satisfy the condition $\lceil u/st \rceil \neq \lceil v/st \rceil$, by symmetry, $Pr(h_{111} = u, h_{211} = v) = [n(n - st)]^{-1}$ for any such (u, v). Thus (1.3.8) holds.

To prove (iii), it suffices to consider h_{111} and h_{112} because of the exchangeability.

Define

$$B_1 = \{(u, v) | \lceil u/st \rceil \neq \lceil v/st \rceil\}. \tag{1.8.1}$$

It is not difficult to verify that the cardinality of B_1 is $m(m - 1)(st)^2$. Define

$$B_2 = \{(u, v) | \lceil u/st \rceil = \lceil v/st \rceil, \lceil (u - \lfloor u/st \rfloor)/s \rceil \neq \lceil (v - \lfloor v/st \rfloor)/s \rceil\}. \tag{1.8.2}$$

It is not difficult to verify that the cardinality of B_2 is $mt(t - 1)s^2$. Note that, for any $(u, v) \in B_1^c \bigcap B_2^c$, where B_1^c and B_2^c are the compliments of B_1 and B_2 respectively, we have $Pr(h_{111} = u, h_{112} = v) = 0$. This is obvious from the construction of DSLHD because $h(:, 1, :)$ has to be an LHD of mt levels. So we only need to consider (u, v) in B_1 and B_2. For (u, v) in B_1, without loss of generality, consider $Pr(h_{111} = 1, h_{112} = st + 1)$. Recall that a DSPM$(m, s, t)$ can be generated in two steps. We now define some notation for the permutations carried out in the two-step construction of the second method in Section 1.2. Denote the row permutation of the ith row of H as π_H^i, the row permutation of the ith row of C^0 as $\pi_{C^0,row}^i$, the column permutation of the jth column of C^0 as $\pi_{C^0,col}^j$. Let $A_1 = \{\pi_H^1(1) = 1\}$, $A_2 = \{\pi_H^{t+1}(1) = 1\}$, $A_3 = \{\pi_{C^0,row}^1(1) = 1\}$, $A_4 = \{\pi_{C^0,row}^{ts+1}(1) = 2\}$, $A_5 = \{\pi_{C^0,col}^1(1) = 1\}$, and $A_6 = \{\pi_{C^0,col}^2(s - 1) = 1\}$. The event $\{h_{111} = 1, h_{112} = st + 1\}$ is

equivalent to $A_1 \bigcap A_2 \bigcap A_3 \bigcap A_4 \bigcap A_5 \bigcap A_6$. Denote

$$E_1 = \{\pi^1_H(1) = 1\} \bigcap \{\pi^1_{C^0,row}(1) = 1\} \bigcap \{\pi^1_{C^0,col}(1) = 1\}. \tag{1.8.3}$$

$$E_2 = \{\pi^{t+1}_H(1) = 1\} \bigcap \{\pi^{ts+1}_{C^0,row}(1) = 2\} \bigcap \{\pi^2_{C^0,col}(s-1) = 1\}. \tag{1.8.4}$$

Since all the permutations are independent of each other, E_1 and E_2 are independent of each other. Hence $Pr(E_1 \bigcap E_2) = Pr(E_1)Pr(E_2)$. Since E_1 is equivalent to $\{h_{111} = 1\}$ and E_2 is equivalent to $\{h_{112} = st + 1\}$, we have $Pr(h_{111} = 1, h_{112} = st + 1) = Pr(h_{111} = 1)Pr(h_{112} = st + 1)$. From (i), $Pr(h_{111} = 1) = Pr(h_{112} = st + 1) = 1/n$. So we have $Pr(h_{111} = 1, h_{112} = st + 1) = 1/n^2$. Similarly we can obtain the same probability for the other pairs in B_1. Recall the cardinality of B_1 is $m(m-1)(st)^2$. We have thus the probability that a (u, v) comes from B_1 is $m(m-1)(st)^2/n^2 = 1 - 1/m$. Hence the probability that a (u, v) comes from B_2 is $1/m$. Since the cardinality of B_2 is $mt(t-1)s^2$, the probability $Pr(h_{111} = u, h_{112} = v) = 1/m(mt(t-1)s^2) = 1/n(n-ms)$ for any (u, v) in B_2. This concludes the proof.

The technique used to prove (iv) is quite similar to (iii). First denote

$$B_1 = \{(u, v) | \lceil u/st \rceil \neq \lceil v/st \rceil\}, \tag{1.8.5}$$

$$B_2 = \{(u, v) | \lceil u/st \rceil = \lceil v/st \rceil, \lceil (u - \lfloor u/st \rfloor)/s \rceil \neq \lceil (v - \lfloor v/st \rfloor)/s \rceil\}, \tag{1.8.6}$$

$$B_3 = \{(u, v) | \lceil u/st \rceil = \lceil v/st \rceil, \lceil (u - \lfloor u/st \rfloor)/s \rceil = \lceil (v - \lfloor v/st \rfloor)/s \rceil\}. \tag{1.8.7}$$

Define the permutations as those in (iii). Without loss of generality, consider the joint distribution of h_{111} and h_{121}. By a similar argument, we can easily show that for any (u, v) from B_1 or B_2, $Pr(h_{111} = u, h_{121} = v) = 1/n^2$. Since the cardinalities of B_1 and B_2 are $m(m-1)(st)^2$ and $mt(t-1)s^2$, respectively, the probability that a (u, v) comes from either B_1 or B_2 is $[m(m-1)(st)^2 + mt(t-1)s^2]/n^2 = 1 - s/n$. Therefore the probability that a (u, v) comes from B_3 is s/n. Since the cardinality of B_3 is $s(s-1)tm$, $Pr(h_{111} = u, h_{121} = v) = s/n(s(s-1)tm) = 1/[n(n-mt)]$ for any (u, v) in B_3. This concludes the proof. □

1.8.2 Proof of Lemma 1.3.2

Proof. For $r = 1, \ldots, s$ and $c = 1, \ldots, t$, express the (i, k)th entry d_{ik} of D_{rc} as

$$d_{ik} = \frac{b_{ik}st + w_{ik}s - e_{ik} - u_{ik}}{n}, i = 1, \ldots, m, k = 1, \ldots, q. \tag{1.8.8}$$

where $\{b_{1k}, \ldots, b_{mk}\}$ constitute a uniform permutation on $\mathbf{Z}_m - 1$; each w_{ik} is a discrete random variable with the probability mass function

$$Pr(w_{ik} = f) = t^{-1}, f = 1, \ldots, t; \tag{1.8.9}$$

each e_{ik} is a discrete random variable with the probability mass function

$$Pr(e_{ik} = a) = s^{-1}, a = 0, \ldots, s - 1; \tag{1.8.10}$$

where u_{ik} are independent $U[0, 1)$ random variables; and b_{ik}, w_{ik} and u_{ik} are mutually independent. Let $l_{ik} = w_{ik}s - e_{ik}$. From the probability mass functions and the mutual independence, it is easy to see that the probability mass function of l_{ik} is

$$Pr(l_{ik} = l) = (st)^{-1}, l = 1, 2, \ldots, st. \tag{1.8.11}$$

Letting $v_{ik} = l_{ik} - u_{ik}$, d_{ik} in (1.8.8) becomes

$$\frac{b_{ik}}{m} + \frac{v_{ik}}{n}. \tag{1.8.12}$$

Since $\{b_{1k}, b_{2k}, \ldots, b_{mk}\}$ is a uniform permutation on $\mathbf{Z}_m - 1$ and b_{ik} and v_{ik} are mutually independent, it remains to verify that $\frac{v_{ik}}{n}$ is a $U(0, \frac{1}{m}]$ random variable, which is shown as follows. For $x \in (0, 1/m]$, let $x_0 = \lceil nx \rceil$ and note that

$$\begin{aligned} Pr(\frac{v_{ik}}{n} \leq x) &= \frac{1}{st}\sum_{a=1}^{st} Pr(\frac{a - u_{ik}}{n} \leq x) \\ &= \frac{1}{st}[\sum_{a=1}^{x_0-1} Pr(\frac{a - u_{ik}}{n} \leq x) + Pr(\frac{x_0 - u_{ik}}{n} \leq x) + \sum_{a=x_0+1}^{st} Pr(\frac{a - u_{ik}}{n} \leq x]. \end{aligned} \tag{1.8.13}$$

Note that $Pr(\frac{a-u_{ik}}{n} \leq x) = 1$ for $a = 1, \ldots, x_0 - 1$; $Pr(\frac{x_0-u_{ik}}{n} \leq x) = 1 - (x_0 - nx)$; and $Pr(\frac{a-u_{ik}}{n} \leq x) = 0$ for $a = x_0 + 1, \ldots, st$, which simplifies (1.8.13) to mx. Thus $\frac{v_{ik}}{n}$ is a $U(0, \frac{1}{m}]$ random variable. This concludes the proof. □

1.8.3 Proof of Lemma 1.3.3

Proof. From Lemma 1.3.2, we have shown the statistical equivalence of D_{rc} to an ordinary LHD. Hence to prove the statistical equivalence of D_r to an SLHD, we only need to consider the joint distribution of two points from two different slices in D_r. Without loss of generality, consider the joint distribution between $D_{r1}(1,:)$ and $D_{r2}(1,:)$, and use $\mathbf{X}_1$ and $\mathbf{X}_2$ to denote them respectively. Define for $0 \leq z_1, z_2 \leq 1$,

$$\delta_{mt}(z_1, z_2) = \begin{cases} 1, & \lceil mtz_1 \rceil = \lceil mtz_2 \rceil, \\ 0, & otherwise; \end{cases} \tag{1.8.14}$$

and

$$\gamma_m(z_1, z_2) = \begin{cases} 1, & \lceil mz_1 \rceil = \lceil mz_2 \rceil, \\ 0, & otherwise. \end{cases} \tag{1.8.15}$$

From Lemma 1.3.1(iii), it can be shown that the joint density function for $\mathbf{X}_1$ and $\mathbf{X}_2$ is

$$p(x_1, x_2) = \frac{n^{2d}}{n^d(n-m)^d} \prod_{k=1}^{d} \{e_0 - e_1\delta_{mt}(x_1^k, x_2^k) - e_2\gamma_m(x_1^k, x_2^k)\}, \tag{1.8.16}$$

where $e_0 = (t-1)t^{-1}$, $e_1 = 1$ and $e_2 = -t^{-1}$, x_1^k and x_2^k are the kth argument of x_1 and x_2 respectively. Similarly, from Lemma 1(iii) in [47], it can be shown that the joint density function between any two points from two different slices in an SLHD is the same. We thus prove the statistical equivalence of D_r and an SLHD. □

1.8.4 Proof of Theorem 1.3.1

Proof. (i) follows immediately from Lemma 1.3.2 and Theorem 1 in [37]. For (ii), the proof of $var_{LH}(\hat{\eta}) \leq var_{IID}(\hat{\eta})$ is the same as in [47]. To establish the first inequality, note

that, for DSLH, we have

$$\begin{aligned} var_{DSLH}(\hat{\eta}) &= var_{DSLH}(\sum_{i=1}^{c} \lambda_i \hat{\mu}_i) \\ &= \sum_{c=1}^{t} \lambda_c^2 var_{DSLH}(\hat{\mu}_c) + \sum_{c_1=1}^{t} \sum_{c_2=1, c_2 \neq c_1}^{t} \lambda_{c_1} \lambda_{c_2} cov_{DSLH}[f^{(c_1)}(\mathbf{d}_1^{c_1}), f^{(c_2)}(\mathbf{d}_2^{c_2})] \\ &= var_{LH}(\hat{\eta}) + \sum_{c_1=1}^{t} \sum_{c_2=1, c_2 \neq c_1}^{t} \lambda_{c_1} \lambda_{c_2} cov_{DSLH}[f^{(c_1)}(\mathbf{d}_1^{c_1}), f^{(c_2)}(\mathbf{d}_2^{c_2})]. \end{aligned} \tag{1.8.17}$$

The last equality follows from the fact that each small piece in a DSLHD is statistically equivalent to an ordinary LHD with the same number of levels. The covariance terms can be divided into two groups. The first group has terms from the same row of a DSLHD. The second group has terms from different rows of a DSLHD. The covariance terms in the same row of a DSLHD are non-positive from the proof of Theorem 1 in [47] and the statistical equivalence of a DSLHD row to an SLHD. For the covariance terms from different rows, similar to the proof of Lemma 1 in [46], we can prove their nonpositiveness. The routine details are omitted. This concludes the proof. □

1.8.5 Proof of Theorem 1.3.2

Proof. From Lemma 1.3.2, each slice of a DSLHD is statistically equivalent to an ordinary LHD. From Lemma 2 in [47], a slice of an SLHD is statistically equivalent to an ordinary LHD. Part (i) then follows immediately. The proof for (ii) is very similar to the proof in [47] and is thus omitted. □

1.8.6 Proof of Lemma 1.4.1

Proof. First, it is obvious that $E(\hat{\mu}^{LHD}) = E(\hat{\mu}^{SLHD}) = \mu$. So it suffices to show that $var(\hat{\mu}^{LHD}) = var(\hat{\mu}^{SLHD})$. Let τ^{LHD} denote the permutation on $\{1, \ldots, n\}$ corresponding to an ordinary LHD, and τ^{SLHD} denote the permutation on $\{1, \ldots, n\}$ corresponding to an SLHD. We have

$$\begin{aligned} var(\hat{\mu}^{LHD}) &= var_{\tau^{LHD}}(E(\hat{\mu}^{LHD}|\tau^{LHD})) + E_{\tau^{LHD}}(var(\hat{\mu}^{LHD}|\tau^{LHD})) \\ &= E_{\tau^{LHD}}(var(\hat{\mu}^{LHD}|\tau^{LHD})), \end{aligned} \tag{1.8.18}$$

where the second equality holds because $E(\hat{\mu}^{LHD}|\tau^{LHD}) = \mu$ is not a function of τ^{LHD}. For $E_{\tau^{LHD}}(var(\hat{\mu}^{LHD}|\tau^{LHD}))$, we first note that

$$\begin{aligned} var(\hat{\mu}^{LHD}|\tau^{LHD}) &= var(\frac{1}{n}\sum_{i=1}^{n} f(x_i^{LHD})|\tau^{LHD}) \\ &= \frac{1}{n}\sum_{i=1}^{n} var(f(\frac{\tau_i^{LHD} - u_i}{n})|\tau^{LHD}) \\ &= \frac{1}{n}\sum_{i=1}^{n} var(f(\frac{i - u_i}{n})), \end{aligned} \tag{1.8.19}$$

where τ_i^{LHD} is the ith component of τ^{LHD} and u_is are independent $U[0,1)$ random variables. The second equality holds because u_is are independent of each other. The third equality holds because u_is are independent of τ^{LHD} and u_is have the same marginal distribution. Finally, we have $var(\mu^{LHD}) = \frac{1}{n}\sum_{i=1}^{n} var(f(\frac{i-u_i}{n}))$, where u_is are i.i.d. $U[0,1)$ random variables. Similarly, we have $var(\mu^{SLHD}) = \frac{1}{n}\sum_{i=1}^{n} var(f(\frac{i-u_i}{n}))$. Hence $var(\mu^{LHD}) = var(\mu^{SLHD})$. □

1.8.7 Proof of Proposition 1.4.1

Proof. First, it is obvious that $E(\hat{\mu}^{LHD}) = E(\hat{\mu}^{SLHD}) = \mu$. So it suffices to show $var(\hat{\mu}^{LHD}) = var(\hat{\mu}^{SLHD})$. For $var(\hat{\mu}^{LHD})$, we have

$$\begin{aligned} var(\hat{\mu}^{LHD}) &= var(\frac{1}{n}\sum_{i=1}^{n} f(\mathbf{x}_i^{LHD})) \\ &= var(\frac{1}{n}\sum_{i=1}^{n}\sum_{j=1}^{d} f_j(x_{i(j)}^{LHD})) \\ &= \frac{1}{n^2}\sum_{j=1}^{d} var(\sum_{i=1}^{n} f_j(x_{i(j)}^{LHD})), \end{aligned} \tag{1.8.20}$$

where $x_{i(j)}^{LHD}$ is the jth component of $\mathbf{x}_i^{LHD}$. The third equality holds because $x_{i(j)}$ and $x_{i(k)}$ are independent of each other for $i = 1,\ldots,n$, $j,k = 1,\ldots,d$ and $j \neq k$. Similarly for

$var(\hat{\mu}^{SLHD})$, we have

$$\begin{aligned} var(\hat{\mu}^{SLHD}) &= var(\frac{1}{n}\sum_{i=1}^{n} f(\mathbf{x}_i^{SLHD})) \\ &= var(\frac{1}{n}\sum_{i=1}^{n}\sum_{j=1}^{d} f_j(x_{i(j)}^{SLHD})) \\ &= \frac{1}{n^2}\sum_{j=1}^{d} var(\sum_{i=1}^{n} f_j(x_{i(j)}^{SLHD})). \end{aligned} \tag{1.8.21}$$

Since each f_j is a function whose argument is uniformly distributed on $(0, 1]$, we have, from Lemma 1.4.1, $var(\sum_{i=1}^{n} f_j(x_{i(j)}^{LHD})) = var(\sum_{i=1}^{n} f_j(x_{i(j)}^{SLHD}))$ for $j = 1, \ldots, d$. This together with (1.8.20) and (1.8.21) implies that $var(\hat{\mu}^{LHD}) = var(\hat{\mu}^{SLHD})$. □

1.8.8 Derivation of (1.5.8)

Define

$$\begin{aligned} B_1 &= \{(u, v)|\lceil u/st\rceil \neq \lceil v/st\rceil\}, \\ B_2 &= \{(u, v)|\lceil u/st\rceil = \lceil v/st\rceil, \lceil (u - \lfloor u/st\rfloor)/s\rceil = \lceil (v - \lfloor v/st\rfloor)/s\rceil)\}, \\ B_3 &= \{(u, v)|\lceil u/st\rceil = \lceil v/st\rceil, \lceil (u - \lfloor u/st\rfloor)/s\rceil \neq \lceil (v - \lfloor v/st\rfloor)/s\rceil)\}. \end{aligned}$$

From Lemma 1.3.1 (iv), $Pr((u, v) \in B_1) = 1 - \frac{1}{m}$, $Pr((u, v) \in B_2) = \frac{1}{mt}$, and $Pr((u, v) \in B_3) = \frac{1}{m} - \frac{1}{mt}$. Let $\delta_{mt}(x_1, x_2)$, $\delta_m(x_1, x_2)$ and $\delta_n(x_1, x_2)$ be similarly defined as in Proposition 1.5.1. The event $(u, v) \in B_1$ is equivalent to $\delta_m(x_1, x_2) = 0$, where $x_1 = \frac{u-u_1}{n}$, $x_2 = \frac{v-u_2}{n}$ and u_1, u_2 are two $U[0, 1)$ random variables. Let X_1 and X_2 be defined as in Proposition 1.5.2. Let $p_1(x_1, x_2)$ denote the joint density function when $\delta_m(x_1, x_2) = 0$. Note that $p_1(x_1, x_2)$ is a constant since u_1 and u_2 are $U[0, 1)$ random variables. Let $A_1 = \{(x_1, x_2) \in (0, 1]^2|\delta_m(x_1, x_2) = 0\}$. We have

$$\int\int_{A_1} p_1(x_1, x_2)dx_1dx_2 = \frac{n(n - st)}{n^2}p_1(x_1, x_2) = 1 - \frac{1}{m}. \tag{1.8.22}$$

Hence $p_1(x_1, x_2) = 1$. Similarly let $A_2 = \{(x_1, x_2) \in (0, 1]^2|\delta_{mt}(x_1, x_2) = 1, \delta_n(x_1, x_2) = 0\}$, and $A_3 = \{(x_1, x_2) \in (0, 1]^2|\delta_m(x_1, x_2) = 1, \delta_{mt}(x_1, x_2) = 0\}$. A_2 and A_3 are equivalent to

B_2 and B_3 respectively with the relationship $x_1 = \frac{u-u_1}{n}$ and $x_2 = \frac{u-u_2}{n}$, where u_1 and u_2 are $U[0, 1)$ random variables. Let $p_2(x_1, x_2)$ denote the joint density function for $(x_1, x_2) \in A_2$, $p_3(x_1, x_2)$ denote the joint density function for $(x_1, x_2) \in A_3$. We have

$$\int\int_{A_2} p_2(x_1, x_2)dx_1dx_2 = \frac{n(s-1)}{n^2}p_2(x_1, x_2) = \frac{1}{mt}, \tag{1.8.23}$$

and

$$\int\int_{A_3} p_3(x_1, x_2)dx_1dx_2 = \frac{n(st-s)}{n^2}p_3(x_1, x_2) = \frac{1}{m} - \frac{1}{mt}. \tag{1.8.24}$$

Hence $p_2(x_1, x_2) = \frac{s}{s-1}$ and $p_3(x_1, x_2) = 1$. From (1.8.22), (1.8.23) and (1.8.24), (1.5.8) holds.

1.8.9 Simulation illustration of Lemma 1.3.1

In this subsection, we use a specific example to illustrate Lemma 1.3.1. Suppose we want to run an experiment of 27 runs. We use a DSLHD for the experiment. Let $s = 3$, $t = 3$, and $m = 3$. To verify the results in Lemma 1.3.1, we randomly generate one million DSLHD(3, 3, 3, 1)s and count the empirical probabilities for each item in the lemma. Based on the exchangeability and symmetry argument, we only count for some of the elements in the H matrix. The details are in Table 1.8.1.

Table 1.8.1: Simulation illustration of Lemma 1.3.1

Definition	Formula	Empirical Value	Theoretical Value
$Pr(h_{111} = 1)$	$1/n$	0.0371	0.0370
$Pr(h_{111} = 1, h_{211} = st + 1)$	$1/[n(n - st)]$	0.0020	0.0021
$Pr(h_{111} = 1, h_{112} = s + 1)$	$1/[n(n - sm)]$	0.0021	0.0021
$Pr(h_{111} = 1, h_{112} = st + 1)$	$1/n^2$	0.0014	0.0014
$Pr(h_{111} = 1, h_{121} = st + 1)$	$1/n^2$	0.0014	0.0014
$Pr(h_{111} = 1, h_{121} = s)$	$1/[n(n - mt)]$	0.0020	0.0021
$Pr(h_{111} = 1, h_{121} = s + 1)$	$1/n^2$	0.0014	0.0014

CHAPTER II

DSLHD-BASED SEQUENTIAL DESIGNS FOR COMPUTER EXPERIMENTS

2.1 Introduction

Computer models representing a physical process are widely used in many areas these days thanks to the advancement of modern computing facilities and algorithms. A computer experiment refers to the mathematical modeling using computer simulation. Computer experiments are used when physical experiments are too expensive or impossible to run. However, computer simulations can still be time-consuming or costly. It can be expedient to use meta-models to supplement computer simulations. Among the meta-models, Gaussian process modeling ([56], [55]) has been successfully applied in many cases and is now standard practice. Meta-models are built on sampled data of computer simulations. To make meta-models more efficient as supplements to computer simulations, we need to consider how to select inputs at which to run computer simulations. This is referred to as experimental design for computer experiments.

There is a large amount of literature concerned with one stage collection of design points ([37]; [27]; [20]). The common and essential feature of these methods is the *space-filling property*, i.e., they all spread points evenly in the design space so that the experimenter will be able to gain information of the underlying model throughout the design space. One stage designs are passive because there is no "learning" from the data collected. A more active method of data selection is the sequential way, which uses a model and some criterion to drive the collection of data. Sequential design in this context has received a lot of attention in the literature ([38]; [4]; [54]; [55]; [28]). In the majority of the proposed sequential strategies, the addition of new data points is done in a fully sequential

manner, i.e., one point per iteration. In some situations, it is more desirable to use batch sequential designs, i.e., add more than one point per iteration. Batch sequential designs are more difficult to conduct because it is more complicated to extend the evaluation of the criterion from fully sequential to batch sequential, especially with the complicated Gaussian process modeling involved. They are more popular nowadays because of the rapid development in parallel computing. Suppose the code takes one day per run and one deploys the fully sequential strategy. The investigator would have to wait for one month just to complete 30 runs. This situation will get worse if the design space is high-dimensional and a relatively large number of data points are needed for model-fitting. On the other hand, it is common to use parallel computing to produce sets of runs at the same time from the code. Imagine we have 30 processors. It takes only one day instead of one month to produce 30 runs if we use the processors all at once, which would result in a dramatic reduction of the turnaround time. The question is: how to select the input values to run a batch sequential experiment.

In recent years, some very interesting space-filling designs, motivated by real problems in computer experiments, have been proposed. More specifically, to deal with modeling in the presence of both quantitative and qualitative factors ([50]), sliced space-filling designs ([49]; [47]) have been proposed. To tackle the problem with more than one level of accuracy of the computer model ([29]; [51]), nested space-filling designs along with their construction have been proposed ([48]). Apart from the space-filling property, these designs also have good sampling properties ([46]; [49]; [47]). Moreover, sliced designs and nested designs can be partitioned into smaller pieces, each of which has the space-filling property. A sequential strategy based on nested Latin hypercube designs has been proposed for modeling high-accuracy and low-accuracy computer models simultaneously ([74]). It was noted in their paper that the design size has to double in each iteration, which makes it inflexible to apply in practice. In this paper, we propose a batch sequential strategy based on doubly sliced Latin hypercube designs (DSLHD), with each batch being a slice of a

DSLHD. A DSLHD is a special LHD that can be partitioned into slices of smaller LHDs, each of which can be further partitioned into slices of even smaller LHDs. The definition and construction of DSLHD was given in the previous chapter. A sliced Latin hypercube design(SLHD) ([47]) is a special case of a DSLHD. Therefore DSLHD is more flexible than SLHD in terms of the batch size at each iteration. This will be elaborated later.

DSLHD-based sequential designs have at least the following advantages. First, the size of the batch at each iteration is quite flexible. It can either be determined by the specific problem, e.g., the number of available computer processors, or from a purely statistical perspective. Secondly, DSLHD has good space-filling properties. Since each slice of a DSLHD is an ordinary LHD, the batch at each iteration of the proposed procedure is a space-filling design. Suppose we are at an early stage of a sequential procedure. We usually do not have a reliable meta-model at this stage because of the paucity of data. This can lead to low quality of the next batch chosen based on the current fitted meta-model. By contrast, the batches in the proposed procedure are always LHDs. We thus avoid choosing batches of low quality since LHDs are space-filling. We call this property of the DSLHD-based sequential designs *robustness to model-fitting*. Thirdly, we can easily bring additional criteria into the sequential procedure to make it more efficient. Such criteria include distance-based criteria ([27]), *IMSPE*([55]) and so on. In this chapter, we also propose a new criterion called expected cross validation error (ECVE). The details are given later. For the optimization based on these criteria, exchange algorithms proposed and developed in the literature ([44]; [32]; [75]; [26]) can be utilized to search for optimal batches.

The remainder of the chapter is organized as follows. In Section 2.2, we briefly review DSLHDs introduced in the previous chapter. More details can be found in Chapter 1. We then introduce the proposed sequential procedure in Section 2.3. Some algebraic results regarding DSLHDs are given in Section 2.4, which provide a guidance on how to choose the parameters in the exchange algorithm. In Section 2.5, we review some existing design criteria and introduce the ECVE criterion. We then present the exchange algorithm for

choosing an optimal slice at each iteration of the sequential procedure. Some numerical studies are presented in Section 2.6. We then present an interesting application of the proposed method to a data center example in Section 2.7. Summary and future research directions are given in Section 2.8.

2.2 A brief introduction to the class of DSLHDs

A DSLHD is a special LHD that can be partitioned into slices of smaller LHDs, which can be further partitioned into slices of even smaller LHDs. For positive integers m, s, t, and d, a DSLHD $\mathbf{D}$ associated with these parameters is an LHD with mst levels in d dimensions. Moreover, $\mathbf{D}$ can be partitioned into $\mathbf{D}_{ij}$s as follows:

$$\begin{array}{cccc} \mathbf{D}_{11} & \mathbf{D}_{12} & \cdots & \mathbf{D}_{1t} \\ \mathbf{D}_{21} & \mathbf{D}_{22} & \cdots & \mathbf{D}_{2t} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{D}_{s1} & \mathbf{D}_{s2} & \cdots & \mathbf{D}_{st} \end{array},$$

where $\mathbf{D}_{ij}$ is a d-dimensional LHD with m runs for each $i = 1, \ldots, s,\ j = 1, \ldots, t$, $\mathbf{D}_i = \cup_{j=1}^{t} \mathbf{D}_{ij}$ is a d-dimensional LHD for each $i = 1, \ldots, s$. For more details of the definition and construction of a DSLHD, see Section 1.2 in Chapter 1.

2.3 A sequential approach

Suppose we are interested in gaining information on an unknown function $y(\mathbf{x}) : [0, 1]^d \to \mathbf{R}$, where $\mathbf{x}$ is uniformly distributed in $[0, 1]^d$. We sample n locations $\mathbf{x}_1, \ldots, \mathbf{x}_n$, run the computer code at these locations and get $y(\mathbf{x}_1), \ldots, y(\mathbf{x}_n)$. A statistical model is then built on $(\mathbf{x}_1, y(\mathbf{x}_1)), \ldots, (\mathbf{x}_n, y(\mathbf{x}_n))$. Our objective is to ensure the statistical model approximates the true function well. Criteria such as the *Integrated Mean Squared Prediction Error* (IMSPE) and the *Maximum Mean Squared Prediction Error* (MMSPE) ([55]) are good examples of evaluating the closeness of the statistical model to the true function. In practice, since we only have information in the n sampled points, we usually use *the cross validation error*

(CVE) to approximate the IMSPE. Hence CVE is a good measure on when we should stop the sequential design. We are now ready to explain the sequential procedure. For positive integers n, m, s, t and d such that $n = mst$, generate a d-dimensional DSLHD $D(m, s, t, d)$, which can be partitioned into $\mathbf{D}_{ij}$s as in Section 2.2. Note that the sequential procedure consists of at most st iterations since we have a budget of $n = mst$ runs and the batch size is m. The proposed procedure is given as follows.

Initialization: Let $\mathbf{D}^{(0)} = \emptyset$ and $\mathbf{D}^{(1)} = D_{11}$, where $\emptyset$ denotes the empty set.

While $k \leqslant st$, do the following:

Step 1. Run the code with $\mathbf{D}^{(k)} \setminus \mathbf{D}^{(k-1)}$.

Step 2. Fit a statistical model $\hat{y}$ for y based on the data $\{(\mathbf{x}_i, y(\mathbf{x}_i))\} : \mathbf{x}_i \in \mathbf{D}^{(k)}\}$.

Step 3. Compute the cross validation error

$$\text{CVE} = \frac{1}{mk} \sum_{\mathbf{x}_i \in \mathbf{D}^{(k)}} \left[\hat{y}_{-\mathbf{x}_i}(\mathbf{x}_i) - y(\mathbf{x}_i)\right]^2 \tag{2.3.1}$$

to assess the accuracy of $\hat{y}$, where $\hat{y}_{-\mathbf{x}_i}$ is the prediction of y based on the data with $\mathbf{D}^{(k)} \setminus \{\mathbf{x}_i\}$.

Step 4. If the CVE is less than a pre-specified threshold e, stop the procedure. Otherwise, let $\mathbf{D}^{(k+1)} = \mathbf{D}^{(k)} \cup \mathbf{D}_{i_0 j_0}$, where $i_0 = \lfloor k/t \rfloor$ and $j_0 = k - i_0 t$.

Note that the initial design and the design at each iteration of the sequential procedure are random. So we can generate a DSLHD first and run the slices sequentially. For the optimized sequential designs considered in Section 2.5, we cannot generate the DSLHD first. Instead, we use the exchange algorithm to search for the optimal slice based on a given criterion and also to retain the DSLHD structure.

2.4 Some algebraic results for doubly sliced Latin hypercube designs

In this section, we calculate the cardinality of the set of DSLHDs for given parameters m, s, t and d. We define two DSLHDs to be different if their permutation matrices are different. For the definition of permutation matrix, see Section 1.2 in Chapter 1. The cardinality measures the difficulty of finding an optimal design at each iteration of the sequential procedure.

Lemma 2.4.1. *There is a total number of* $(t!)^{md-1}$ *SLHDs on* $(0,1]^d$*, where t is the number of slices of the SLHD and m is the number of runs per slice.*

Based on Lemma 2.4.1, we now calculate the cardinality of the set of DSLHDs for given parameters.

Proposition 2.4.1. *There is a total number of* $((s!)^{mt-1}(t!)^{m-1})^d(s!(t!)^s)^{d-1}$ *DSLHD(m, s, t, d)s on* $(0,1]^d$.

Since the design is constructed sequentially, it is useful to know, at each iteration of the procedure, the number of admissible designs within the DSLHD framework. A more detailed description of the sequential procedure is needed for the calculation of admissible designs at each iteration of the procedure. Suppose the sequential procedure is based on a DSLHD(m, s, t, d). Let $\mathbf{Z}_n = \{1, 2, \ldots, n\}$. Partition $\mathbf{Z}_n$ into $Z'_{ij}s$ such that

$$Z_{ij} = \{(i-1)st + (j-1)s + 1, (i-1)st + (j-1)s + 2, \ldots, (i-1)st + js\}, \qquad (2.4.1)$$

where $i = 1, \ldots, m$, $j = 1, \ldots, t$. Because we have a budget of $n = mst$ runs and the batch size is m, we have at most st iterations for the sequential design. While $k \leq st$, do the following starting with $k = 0$:

If $k - \lfloor k/t \rfloor = 0$, each Z_{ij} is left with $s - k/t$ elements. For i from 1 to m, randomly sample one piece Z_{in_i} from $Z_{i1}, \ldots, Z_{it}$, then randomly sample one point from each sampled piece Z_{in_i}. Do this for each dimension. Denote the sampled points for the

pth dimension as $x^p_{k1}, \ldots, x^p_{km}$. Fix the order for the first dimension, do a random permutation for each of the other $d-1$ dimensions. We then get the permutation matrix of the design at the kth iteration.

If $k - \lfloor k/t \rfloor > 0$, for each i from 1 to m, $t - k + \lfloor k/t \rfloor$ of the $Z'_{ij}s$ are left with $s - \lfloor k/t \rfloor$ elements and $k - \lfloor k/t \rfloor$ of them are left with $s - \lfloor k/t \rfloor - 1$ elements. For i from 1 to m, randomly select one piece from those $Z'_{ij}s$ that have $s - \lfloor k/t \rfloor$ elements, denote this piece as Z_{in_i}, then randomly sample one point from each sampled piece Z_{in_i}. Do this for each dimension. Assume the sampled points for the pth dimension are $x^p_{k1}, \ldots, x^p_{km}$. Fix the order for the first dimension, do a random permutation for each of the other $d-1$ dimensions. We then get the permutation matrix of the design at the kth iteration.

We are now ready to present results on the number of admissible designs at each iteration of the sequential procedure.

Proposition 2.4.2. *At the kth iteration of the sequential procedure, the total number of admissible designs is*

$$\begin{cases} ((s - k/t)t)^{md}(m!)^{d-1}, k - \lfloor k/t \rfloor = 0, \\ ((s - \lfloor k/t \rfloor)(t - k + \lfloor k/t \rfloor))^{md}(m!)^{d-1}, k - \lfloor k/t \rfloor > 0. \end{cases} \tag{2.4.2}$$

Proposition 2.4.2 calculates the number of admissible designs at each iteration of the sequential procedure. This result will be useful for choosing optimized sequential design, which will be addressed in the next section.

2.5 Optimized sequential design

In the sequential procedure introduced in Section 2.3, the design was chosen randomly at each iteration. In this section, we present optimized sequential design based on some criterion. We first review some existing criteria in the literature and then propose a new criterion called expected cross validation error (ECVE). As will be seen in the numerical experiments, the optimized sequential design is more efficient than the random sequential

design. The optimization at each iteration is done using the enhanced stochastic evolutionary algorithm ([26]). Note we are using the algorithm in a sequential framework, while the algorithm was originally used to choose optimal one-stage design. Since we need to keep the DSLHD structure, our updating scheme is different from [26] and more involved. More details are in Section 2.5.3.

2.5.1 A review of some existing criteria

In this section, we give a short review of some existing criteria for selecting a design. The first criterion is maximin distance criterion ([27]). This is based on a measure or metric that quantifies how spread out a set of points are. For a design set D, define the smallest distance between any two points in D to be

$$\min_{\mathbf{x}_1,\mathbf{x}_2\in D} \rho_p(\mathbf{x}_1,\mathbf{x}_2), \tag{2.5.1}$$

where $\rho_p(\mathbf{x}_1,\mathbf{x}_2) = [\sum_{i=1}^{d} |x_{1i} - x_{2i}|^p]^{1/p}$. For a design class $\mathcal{D}$, the design that maximizes (2.5.1) is said to be a *maximin distance design* and denoted by D_{Mm}; thus

$$\min_{\mathbf{x}_1,\mathbf{x}_2\in D_{Mm}} \rho_p(\mathbf{x}_1,\mathbf{x}_2) = \max_{D\in\mathcal{D}} \min_{\mathbf{x}_1,\mathbf{x}_2\in D} \rho_p(\mathbf{x}_1,\mathbf{x}_2). \tag{2.5.2}$$

Maximin distance criterion is one of several well-known distance-based criteria. The others include minimax criterion ([27]), average distance criterion and so on. Since we only use maximin distance criterion in the chapter, we omit the details of the other criteria.

The second criterion is maximum entropy criterion. We review this criterion in the context of Gaussian process modeling, which is commonly used in computer experiments. A Gaussian process model is defined as follows:

$$Y(\mathbf{x}) = \mathbf{f}(\mathbf{x})^T\beta + Z(\mathbf{x}), \tag{2.5.3}$$

where $\mathbf{f}(\mathbf{x}) = (f_1(\mathbf{x}), \ldots, f_p(\mathbf{x}))$ is a pre-specified set of functions, β is a vector of unknown regression coefficients, and $Z(\mathbf{x})$ is a Gaussian process with mean zero, variance σ^2 and

correlation function R parameterized by a set of correlation parameters $\Psi = (\psi_1, \ldots, \psi_d)^T$, d is the dimension of $\mathbf{x}$. The covariance function is defined as

$$cov(Y(\mathbf{x}_1), Y(\mathbf{x}_2)) = \sigma^2 R(\mathbf{x}_1 - \mathbf{x}_2|\Psi). \tag{2.5.4}$$

A popular choice for R is the Gaussian correlation function

$$R(\mathbf{x}_1 - \mathbf{x}_2|\Psi) = exp(-\sum_{k=1}^{d} \psi_k (x_{1k} - x_{2k})^2). \tag{2.5.5}$$

For the Gaussian process models under consideration, the training data has the conditional distribution

$$\mathbf{Y}^n|\beta \sim N(\mathbf{F}\beta, \sigma^2 R), \tag{2.5.6}$$

where $\mathbf{F}$ is the $n \times p$ matrix of regressors having (i,j)th element $f_j(\mathbf{x}_i)$. One can show that a maximum entropy design maximizes the determinant of the variance of the observed responses at the points in the design. If we assume a Gaussian prior for β

$$\beta \sim N_p(\mathbf{b}_0, \tau^2 V_0), \tag{2.5.7}$$

the determinant of the marginal covariance matrix of the vector of observations $\mathbf{Y}^n$ is

$$\begin{aligned} & det(\sigma^2 R + \tau^2 \mathbf{F} V_0 \mathbf{F}^T) \\ = \; & det(\sigma^2 R) det(\tau^2 V_0 \mathbf{F}^T (\sigma^2 R)^{-1} \mathbf{F} + \mathbf{I}_p). \end{aligned} \tag{2.5.8}$$

If β is treated as fixed, i.e., $\tau^2 = 0$, the maximum entropy criterion reduces to

$$det(\sigma^2 R). \tag{2.5.9}$$

2.5.2 The expected cross validation error criterion

In this section, we propose a new criterion called ECVE. Suppose, at the ith iteration, we have sampled data $(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_n, y_n)$. Let Y_n represent the vector $(y_1, \ldots, y_n)^T$. Suppose the batch size is m. Denote the m runs in an unsampled batch as $\mathbf{x}_{n+1}, \ldots, \mathbf{x}_{n+m}$. Define the expected cross validation error as

$$E(\sum_{i=1}^{n+m} (y_{-\mathbf{x}_i}(\mathbf{x}_i) - y(\mathbf{x}_i))^2 | Y_n). \tag{2.5.10}$$

Given $\mathbf{x}_1, \ldots, \mathbf{x}_n$ and $y_1, \ldots, y_n$, we can estimate the unknown parameters β and Ψ using maximum likelihood estimation, assuming the Gaussian process model in (2.5.3). The details can be found in [58]. For an unsampled point $\mathbf{x}_0$, the predicted value $\hat{y}(\mathbf{x}_0)$ is

$$\hat{y}(\mathbf{x}_0) = \mathbf{f}_0^T\hat{\beta} + \mathbf{r}_0^T\mathbf{R}^{-1}(Y_n - \mathbf{F}\hat{\beta}), \tag{2.5.11}$$

where $\hat{\beta} = (\mathbf{F}^T\mathbf{R}^{-1}\mathbf{F})^{-1}\mathbf{F}^T\mathbf{R}^{-1}Y_n$. Here $\mathbf{f}_0 = \mathbf{f}(\mathbf{x}_0)$ is the $p \times 1$ vector of regressors at $\mathbf{x}_0$, $\mathbf{F}$ is the $n \times p$ matrix of regressors having (i, j)th elements $\mathbf{f}_j(\mathbf{x}_i)$ for $1 \leq i \leq n, 1 \leq j \leq p$, $\mathbf{r}_0 = (R(\mathbf{x}_0 - \mathbf{x}_1), \ldots, R(\mathbf{x}_0 - \mathbf{x}_n))^n$, and $\mathbf{R}$ is the $n \times n$ correlation matrix of the n sampled observations. Note that the predictor is a linear function of Y_n given the estimates of the unknown parameters. For evaluation of the ECVE criterion, the estimates of the unknown parameters β, ψ are based on the sampled data. Let Y_{n+m} denote the vector $(y_1, \ldots, y_n, y_{n+1}, \ldots, y_{n+m})^T$. Let $\hat{y}_{-\mathbf{x}_i} = a_i^T Y_{n+m}$, where

$$a_i = (a_{i,1}, a_{i,2}, \ldots, a_{i,i-1}, 0, a_{i,i+1}, \ldots, a_{i,n+m}).$$

Hence $\hat{y}_{-\mathbf{x}_i} - y(\mathbf{x}_i) = b_i^T Y_{n+m}$, where $b_i = (a_{i,1}, \ldots, a_{i,i-1}, -1, a_{i,i+1}, \ldots, a_{i,n+m})^T$. Therefore we have

$$\begin{aligned}
& E \quad (\sum_{i=1}^{n+m}(y_{-\mathbf{x}_i}(\mathbf{x}_i) - y(\mathbf{x}_i))^2 | Y_n) \\
&= \quad E(\sum_{i=1}^{n+m} Y_{n+m}^T b_i b_i^T Y_{n+m} | Y_n) \\
&= \quad E(Y_{n+m}^T B Y_{n+m} | Y_n) \\
&= \quad E(tr(Y_{n+m}^T B Y_{n+m}) | Y_n) \\
&= \quad E(tr(B Y_{n+m} Y_{n+m}^T) | Y_n) \\
&= \quad tr(B E(Y_{n+m} Y_{n+m}^T | Y_n)).
\end{aligned} \tag{2.5.12}$$

Let U_m denote $(y_{n+1}, \ldots, y_{n+m})^T$. Then $Y_{n+m}Y_{n+m}^T$ can be written as $\begin{pmatrix} Y_n Y_n^T & Y_n U_m^T \\ U_m Y_n^T & U_m U_m^T \end{pmatrix}$. For $Y_n Y_n^T$, we have

$$E(Y_n Y_n^T | Y_n) = Y_n Y_n^T. \tag{2.5.13}$$

For $Y_n U_m^T$, we have

$$E(Y_n U_m^T | Y_n) = Y_n E(U_m^T | Y_n) = Y_n Y_n C_n^T, \tag{2.5.14}$$

where C_n is a $m \times n$ matrix such that $E(y_{n+i}|Y_n) = C_{ni.} Y_n$ and $C_{ni.}$ is the ith row of C_n. For $U_m U_m^T$, we have

$$\begin{aligned} E(U_m U_m^T | Y_n) &= cov(U_m|Y_n) + E(U_m|Y_n)(E(U_m|Y_n))^T \\ &= R_{m,m} - R_{m,n} R_{n,n}^{-1} R_{n,m} + C_n Y_n Y_n^T C_n^T, \end{aligned} \tag{2.5.15}$$

where $R_{m,m}$ is the covariance matrix of U_m given the estimates of the correlation parameters and the estimate of the variance, $R_{m,n}$ is the covariance matrix between U_m and Y_n, i.e., $R_{m,n} = cov(U_m, Y_n^T)$, $R_{n,m}$ is the transpose of $R_{m,n}$, $R_{n,n}$ is the covariance matrix of Y_n. Combining (2.5.13), (2.5.14) and (2.5.15), (2.5.12) can be written as $tr(BC)$, where

$$C = \begin{pmatrix} Y_n Y_n^T & Y_n Y_n^T C_n^T \\ C_n Y_n Y_n^T & R_{m,m} - R_{m,n} R_{n,n}^{-1} R_{n,m} + C_n Y_n Y_n^T C_n^T \end{pmatrix}. \tag{2.5.16}$$

We have thus derived a closed form for the ECVE criterion, which can be easily implemented.

2.5.3 The algorithm

The optimal design algorithm consists of two parts. The first part is the enhanced stochastic evolutionary algorithm. See [26] for details. The second part is an updating scheme in the exchange algorithm. For ease of explanation and without loss of generality, we focus on one dimension of the design variable in the following discussion. At each iteration of the sequential procedure, the constraint is to keep the DSLHD structure. Suppose we have randomly chosen a slice from a DSLHD. There are two updating operations. The first is to exchange two elements within the chosen slice. It is obvious that this operation keeps the DSLHD structure. The second is to exchange one element in the slice with an element from a different slice. As discussed in Section 2.4, at each iteration of the sequential design, we randomly choose from one of the $Z'_{ij}s$ and then randomly select one point from the

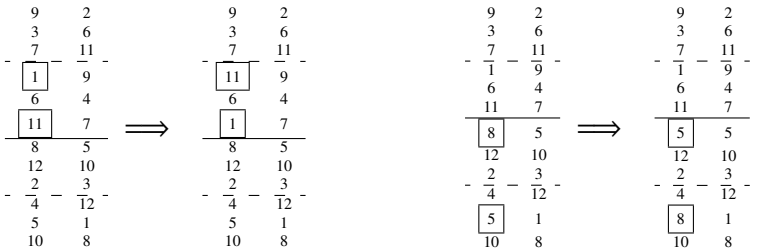

Figure 2.5.1: Illustration of the two updating operations

chosen Z_{ij}. Thus in order to keep the DSLHD structure, we can only exchange the elements between $Z'_{ij}s$ in the *same* row. Moreover, if some slices in the same row have already been sampled in the previous iterations, we can only exchange elements between those $Z'_{ij}s$ that have not been sampled yet. We illustrate the updating schemes using a simple example. Suppose we perform the sequential procedure based on a DSLHD(3,2,2,2). Two examples of the aforementioned updating operations are illustrated in Figure 2.5.1. Both are done for the first dimension. The left side shows an example of the first updating operation. Note that the solid line separates two bigger slices and the dashed lines in each bigger slice separate smaller slices. Entry 1 and 11, being two elements in the same smaller slice, are exchanged. An example of the second updating operation is illustrated on the right side. Here we exchange 8 and 5. These two elements are in two different smaller slices. In order to keep the DSLHD structure, 8 can only be exchanged with 5 when the other two numbers in the same slice as 8 are fixed. This is because we need to make sure that each smaller slice is an LHD and the bigger slice is an LHD. For example, 8 cannot be exchanged with 4.

2.6 Numerical illustration

In this section, we use a few examples to illustrate the proposed sequential procedure. Suppose the batch size is m and the dimension of the design variable is d. We first compare three schemes defined as follows.

Definition 1. *Suppose m, n, t, t_1 and t_2 are positive integers with $n = mt = mt_1t_2$.*

(i). Let LH denote the scheme that generates an ordinary LHD with n levels and randomly divide the generated LHD into t batches and run them sequentially.

(ii). Let SLH denote the scheme that generates an SLHD of t slices, each of which is an ordinary LHD with m levels and run the sequential design with the batch at each iteration being a slice of the SLHD.

(iii). Let DSLH denote the scheme that generates a DSLHD(m, t_1, t_2, d) and run the sequential design with the batch at each iteration being a slice of the DSLHD.

We then compare the random DSLHD-based sequential design and the optimized DSLHD-based sequential design and show the advantage of the latter. In this chapter, we only consider maximin distance criterion for the optimized sequential design.

Example 2.1. The following function in [11] is used:

$$y = [1 - exp(-1/2x_2)]\frac{2300x_1^3 + 1900x_1^2 + 2092x_1 + 60}{100x_1^3 + 500x_1^2 + 4x_1 + 20}.$$

The domain of $\mathbf{x}$ is $[0, 1]^2$. Suppose we have a budget of 72 runs for the design. The parameters for the DSLH scheme are $m = 6$, $t_1 = 4$, $t_2 = 3$. The parameters for the SLH sheme are $m = 6$, $t = 12$. For each of the three schemes, after a permutation matrix is generated, we use the mid-point in each small interval as the design point. At each iteration, a kriging model is fitted and prediction is performed on a predefined regular grid. The mean squared prediction error on the regular grid is calculated at each iteration. This whole procedure is replicated 100 times for each scheme. The mean value and standard deviation of the mean squared prediction error for the three schemes over the 100 replications are shown in Figures 2.6.1 and 2.6.2 respectively.

We have the following observations from Figures 2.6.1 and 2.6.2:

(i). DSLH and SLH are comparable in terms of the average RMSE. LH is worse than these two in earlier iterations. This is expected because each slice of a DSLHD or an SLHD is an ordinary LHD. So at each iteration, the batch from the DSLH scheme or

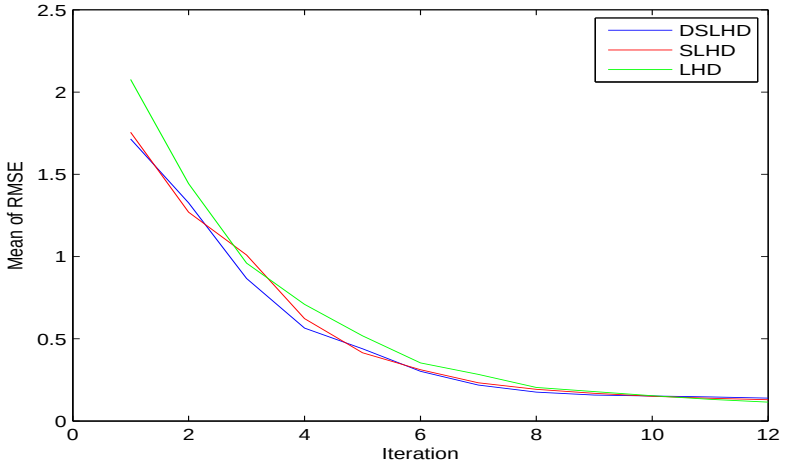

Figure 2.6.1: Average value of RMSE for DSLH, SLH and LH in Example 2.1

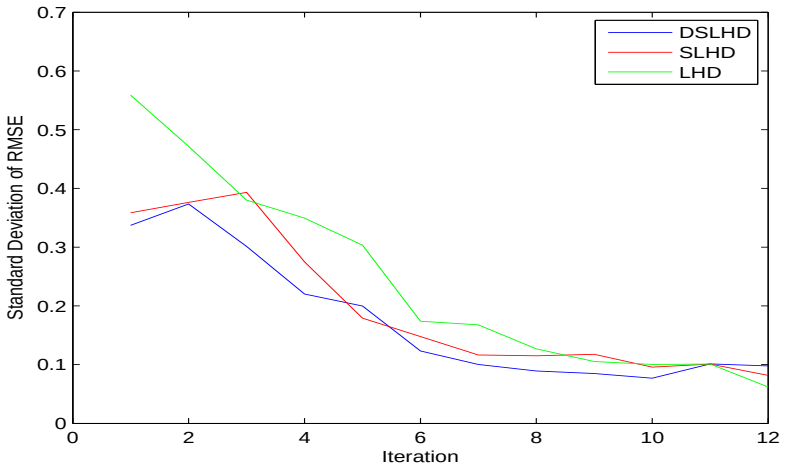

Figure 2.6.2: Standard deviation of RMSE for DSLH, SLH and LH in Example 2.1

the SLH scheme is an ordinary LHD. By contrast, the batch from the LH scheme is a random subset of an ordinary LHD.

(ii). In the last few iterations, the three schemes are comparable in terms of the average RMSE because each of them forms an LHD when viewed as an overall design.

(iii). The standard deviation of RMSE in the LH scheme is higher than those in the other two schemes because each batch in the SLH and DSLH scheme is an ordinary LHD.

By contrast, LH does not have this property.

Next we use this example to illustrate the advantage of optimized sequential design over the random one. The result is shown in Figure 2.6.3. The green line shows the RMSE for the optimized design. The boxplots of RMSE for the random design are based on 100 replications. The RMSE of the optimized design is smaller than the median RMSE of the random design for most of the iterations. Moreover, we can prevent unusually large RMSE by using the optimized design. In practice, we use the cross validation error (CVE) as the

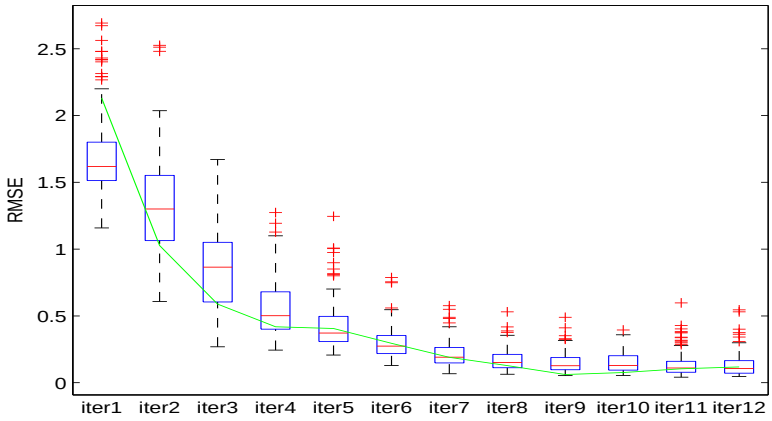

Figure 2.6.3: Comparison of the optimized design and the random design in Example 2.1

criterion to decide when to stop sampling. Hence we also provide a study on the trends of the square root of cross validation error (RCVE) and RMSE. The mean and standard deviation of these two are studied over 100 replications. The results are shown in Figures 2.6.4 and 2.6.5. The trend of RCVE and RMSE agree very well with each other. Note that RCVE has larger standard deviation than RMSE when the sample size is small.

Next we consider a four-dimensional function.

Example 2.2. The following function from [9] is used:

$$y = \frac{x_1}{2}[\sqrt{1 + (x_2 + x_3^2)x_4/x_1^2}] + (x_1 + 3x_4)exp[1 + sin(x_3)].$$

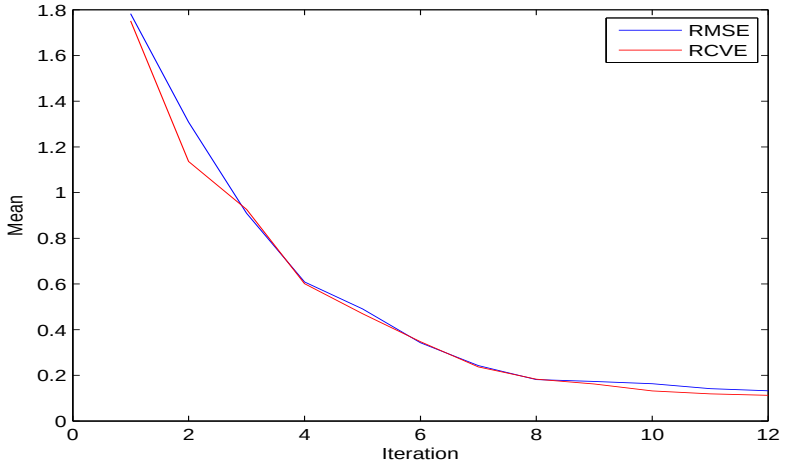

Figure 2.6.4: Average values of RMSE and RCVE in Example 2.1

Table 2.6.1: Inputs and their ranges of the borehole model

Input	Range	Unit
r_w: radius of borehole	[0.05, 0.15)	m
r: radius of influence	[100, 50000)	m
T_u: transmissivity of upper aquifer	[63070, 115600)	m^2/yr
H_u: potentiometric head of upper aquifer	[990, 1110)	m
T_l: transmissivity of lower aquifer	[63.1, 116)	m^2/yr
H_l: potentiometric head of lower aquifer	[700, 820)	m
L: length of borehole	[1120, 1680)	m
K_w: hydraulic conductivity of borehole	[1500, 15000)	m/yr

The domain of $\mathbf{x}$ is $[0,1]^4$. To save space, we only show the comparison between the optimized design based on maximin distance criterion and the random design. Suppose we have a budget of 120 runs. For the random design, we randomly generate 100 DSLHD(10, 6, 2, 4)s and run the sequential procedure based on them. Since the RMSE decreases very quickly with very small standard deviation at each iteration, we only give the mean of RMSE for the random design instead of the box plots. The results are shown in Figure 2.6.6. Again the optimized design consistently outperforms the random design.

Borehole example. We now illustrate the proposed methods with the borehole model ([39]). Table 2.6.1 gives the eight inputs and their ranges and units of the model. This model

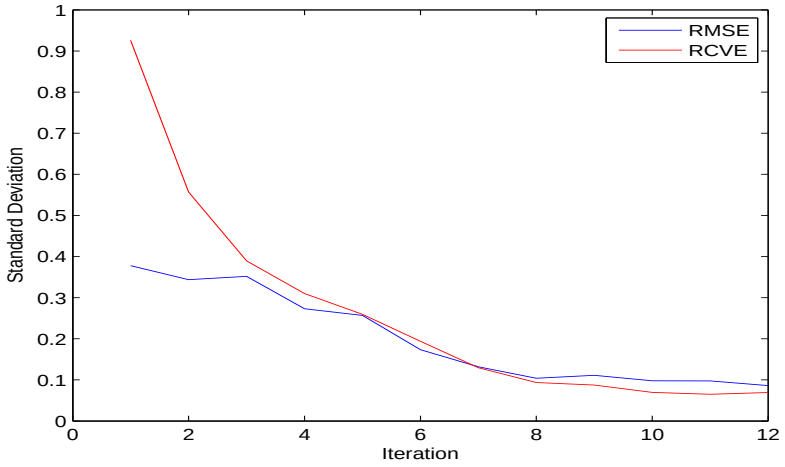

Figure 2.6.5: Standard deviation of RMSE and RCVE in Example 2.1

describes the flow of water through a borehole drilled from the ground surface through two aquifers. The response of the model is given by

$$y = \frac{2\pi T_u (H_u - H_l)}{\log(r/r_w)\Big[1 + \dfrac{2LT_u}{\log(r/r_w) r_w^2 K_w} + T_u/T_l\Big]}.$$

The input variables are scaled so that each of them takes values on the unit interval. Suppose we have a budget of 180 runs. We only compare the random design and the optimized design. For the random design, we generate 100 DSLHD(10, 6, 3, 8)s. The average value of the RMSE of the random design and of the optimized design are shown in Figure 2.6.7. Note these two lines are very close to each other. It shows that the optimized design can achieve the average performance of the random design.

2.7 Subset selection for large data

2.7.1 A different perspective of sequential design

In this section, we illustrate the use of DSLHD-based sequential design from a different perspective. As pointed out by MacKay ([36]), there are two scenarios in which we are able to actively select training data. The first is when the collection of data points is expensive or slow. The second is when we have a large amount of data and we want to select a subset

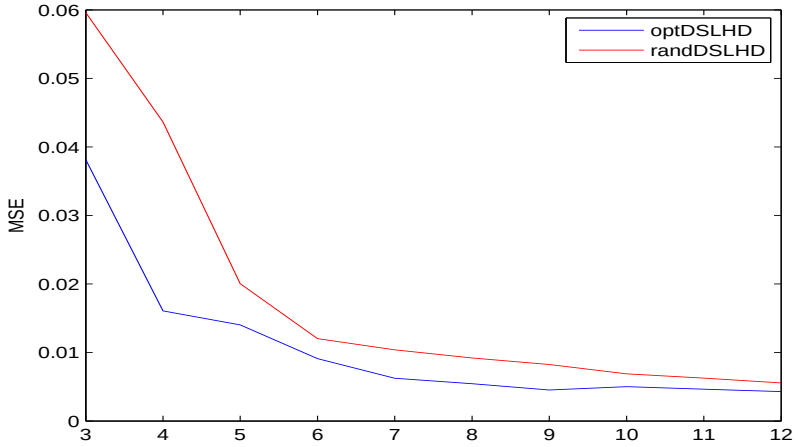

Figure 2.6.6: Comparison of the optimized design and the random design in Example 2.2

of the data that is most informative. There are at least two reasons why we only select a subset in the second scenario. The first consideration is computational efficiency. The second is that more data points do not necessarily lead to more information. For modeling in computer experiments, the Gaussian process model is typically used. Its computational complexity is $O(n^3)$, where n is the sample size. Hence it is computationally intensive to apply the Gaussian process model to large data sets. Moreover, the values of the design variable are typically close to each other in a large data set. The Gaussian process model can easily become numerically unstable in such situations. We use a data center example in the next section to illustrate the discussion here.

2.7.2 Application to a data center example

A data center is a facility used to house computer systems and associated components, such as telecommunications and storage systems ([57]). Data center consumes a lot of power each year, a great amount of which is due to the cooling system in the data center. On the one hand, we want the temperature in the data center to be low so that the computing facilities and storage systems can work stably. On the other hand, we want to lower the cooling cost. It is thus very critical to design an energy-efficient data center. For such

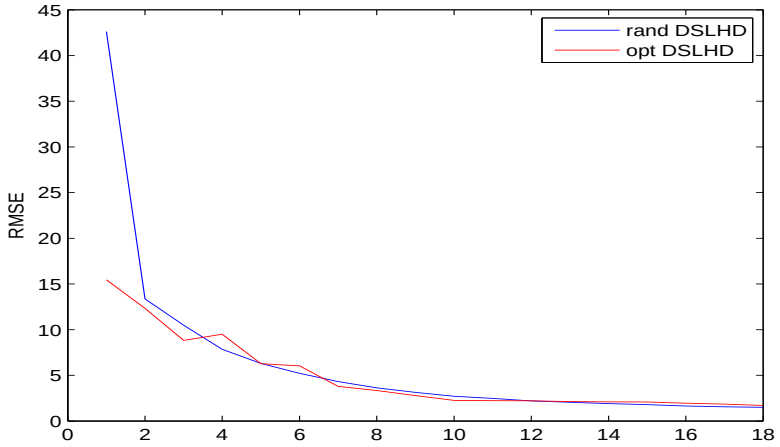

Figure 2.6.7: Comparison of the optimized design and the random design in the borehole example

an engineering design, the first step is to predict the temperature distribution for different cooling settings. In the scenario considered by [57], the cooling setting is represented by the inlet velocity of the computer room air conditioning (CRAC). Due to the high cost of real operations in a data center, computer simulation is used to predict the temperature distribution in the data center under different inlet velocities of CRAC. Computational Fluid Dynamics/Heat Transfer (CFD/HT) is currently used to simulate the temperature distribution in the data center. It is time consuming to use CFD/HT for complex system design and is thus not practical for iterative and optimization-based design methods ([57]). A typical solution to this problem is to build a meta-model based on the simulations and use the meta-model to replace the simulation model. Among the meta-models, the Gaussian process model has been the most successful. Therefore, we utilize it as the meta-model to tackle the computational issue. The data consists of twelve temperature distributions corresponding to twelve different inlet velocities. The spatial locations where the temperature is recorded are the same for different inlet velocities. The objective is to predict the temperature distribution for those inlet velocities that are not in the data set. Let v denote the inlet velocity of the CRAC. Denote the temperature distribution for each v as $T(v, x, y, z)$, where

(x, y, z) is a three-dimensional spatial location. In this application, we are interested in the temperature distribution in one rack. Hence the spatial location reduces to two dimensions, denoted by (x, z). For each v, we have a temperature distribution across 1296 spatial locations. The spatial locations are very close to a regular grid. For illustration, we display the temperature distribution in the rack for $v = 1.0m/s$ in Figure 2.7.1. Denote the twelve

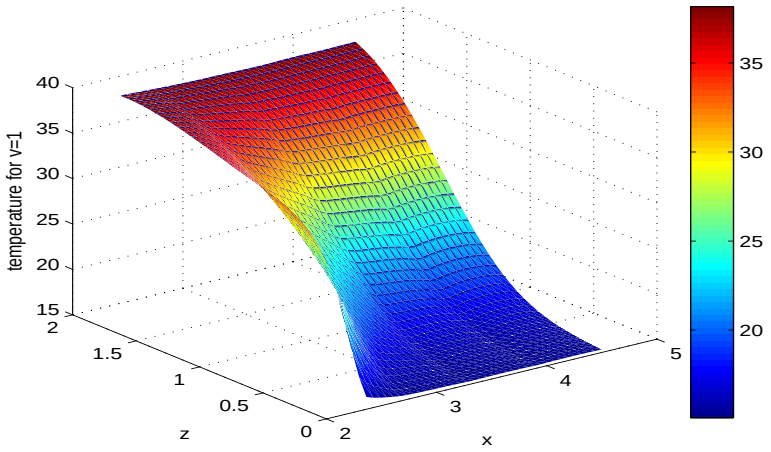

Figure 2.7.1: Temperature distribution in the rack for v=1.0m/s

velocities by $v_1, v_2, ..., v_{12}$, the spatial locations by D and let $V_i = v_i \mathbf{1}_{1296}$, where $\mathbf{1}_{1296}$ is a 1296×1 column vector of ones. The large size of the data makes it impossible to naively apply the Gaussian process model to the whole data set. The question is whether we can use a smaller number of data points to fit the model and get sufficient prediction accuracy. If so, how should we sample from the locations for model-fitting? We propose to use the DSLHD-based sequential design to sample from the locations and build a Gaussian process model at each iteration. Suppose at the kth iteration, the sampled spatial locations are $(x_1, z_1), (x_2, z_2), \ldots, (x_{mk}, z_{mk})$. Use D^k to denote the spatial locations. Let V_i^k be the $mk \times 1$ vector whose values are all v_i. Let $T(V_1^k, D^k), T(V_2^k, D^k), \ldots, T(V_{12}^k, D^k)$ be the temperature values. For $i = 1, \ldots, 12$, leave $T(V_i^k, D^k)$ out and use the remaining data to fit a Gaussian process model. Use the fitted model to predict for $T(V_i, D)$ and denote the predicted value

as $\hat{T}(V_i, D)$. The RCVE at the kth iteration is defined as

$$\text{RCVE} = \frac{1}{12}\sum_{i=1}^{12}\sqrt{\|\hat{T}(V_i, D) - Y(V_i, D)\|^2/1296}. \tag{2.7.1}$$

The procedure stops when the RCVE is less than a pre-specified value. Here we also compare the random design and the optimized design. The results are shown in Figure 2.7.2. The green line is the RCVE for the optimized design. The box plots of the RCVE for the random design is based on 100 replications. Note that the optimized design performs much better than the random design. Based on the input from our collaborators, we can

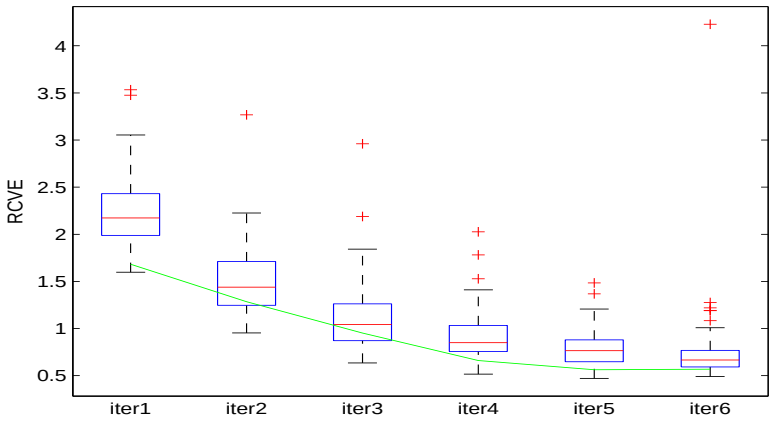

Figure 2.7.2: Comparison of the random design and the optimized design in the data center example

choose 1 degree as the threshold value for RCVE to stop sampling. Hence we can stop at the fourth iteration based on the optimized design. Note that we only need $24 \times 12 = 288$ data points to reach a desirable prediction accuracy.

2.8 Concluding remarks

We have proposed DSLHD-based sequential designs for running computer experiments. This procedure has the property of robustness to model-fitting. The batch at each iteration is an ordinary LHD, which helps to avoid choosing low quality batches at early iterations

of a sequential procedure when the meta-model is not reliable because of the paucity of data points.

It warrants further work to find out when DSLHD-based sequential designs are more efficient than SLHD-based sequential designs. In particular, the question is: how should we take advantage of the doubly sliced structure of DSLHDs to make the sequential procedure more efficient? We will also compare our method with other batch sequential design methods in the literature ([34]).

In this chapter, we have constructed optimized designs based on maximin distance criterion and the newly proposed ECVE criterion. However, the ECVE-based sequential designs do not seem to work better than the distance-based sequential designs. The problem may be due to the fitting of kriging models. Kriging models are very sensitive to the initial values chosen for the parameters. For the exchange algorithm in Section 2.5, the parameters are currently chosen as suggested in [26]. We plan to empirically study how to choose these parameters in our context based on the algebraic results in Section 2.4.

2.9 Appendix

2.9.1 Proof of Lemma 2.4.1

Proof. First consider one dimension of the design variable. Denote the n levels as $1, \ldots, n$. For an SLHD, we partition the n levels into m subsets, where the first subset is $\{1, \ldots, t\}$, the second subset is $\{t+1, \ldots, 2t\}$ and so on. To construct an SLHD, we first randomly choose one value from each of the m subsets to form one slice. There is a total of $(t!)^{m-1}$ possible choices for this step. For a d-dimensional design, the total number of choices is then $(t!)^{(m-1)d}$. After the slices for each dimension have been chosen, we have different combinations across the d dimensions at the slice level. For a fixed choice of slices, all the combinations at the slice level is $(t!)^{d-1}$. Thus the total number of designs is $(t!)^{md-1}$. □

2.9.2 Proof of Proposition 2.4.1

Proof. To construct a DSLHD(m, s, t, d), we first divide the mst levels into s slices, each of which consists of mt levels. From Lemma 2.4.1, there is a total of $(s!)^{mt-1}$ possible choices at this step. After the s slices have been chosen, the next step is to divide each of the s slices into t smaller slices, each of which has m runs. Again from Lemma 2.4.1, there is a total of $(t!)^{m-1}$ choices for each of the s slices. Thus for a DSLHD(m, s, t, d), there is a total of $((s!)^{mt-1}(t!)^{m-1})^d$ choices to divide the n levels into st slices. After the st slices have been chosen for each dimension, the final step is to permutate the st slices for each dimension. The permutation is done in two steps. We first permutate the s bigger slices. There is a total of $s!$ permutations. After the locations of the s slices are fixed, we permutate the t slices in each of the s slices with a total of $(t!)^s$ choices. Hence there is a total of $(s!(t!)^s)^{d-1}$ permutations. Thus we have in total $((s!)^{mt-1}(t!)^{m-1})^d(s!(t!)^s)^{d-1}$ designs. This concludes the proof. □

2.9.3 Proof of Proposition 2.4.2

Proof. We give a proof for $k - \lfloor k/t \rfloor = 0$. When $k - \lfloor k/t \rfloor = 0$, we randomly sample one piece from $\{Z_{ij}\}_{j=1}^{t}$ for each i. There are t possible outcomes for each i. For each i, we then randomly sample one point from the $(s - k/t)$ points of the sampled piece. There are $(s - k/t)$ outcomes for each i. Therefore the total number of choices of element is $(s - k/t)t$ for each i. Since this is done independently for each i, the total number of choices of the m elements is $((s - k/t)t)^m$. It is then easy to prove that the total number of choices of the m elements for all the d dimensions is $((s - k/t)t)^{md}$. After the elements of each dimension have been chosen, we fix the order of the m elements of the first dimension and randomly permutate the elements of the other $d - 1$ dimensions, with the permutation carried out independently for each dimension. It is easy to show that the total number of choices for this step is $(m!)^{d-1}$. Combining the results from the two steps, the total number of choices for the next slice is $((s - k/t)t)^{md}(m!)^{d-1}$ when $k - \lfloor k/t \rfloor = 0$. The proof for $k - \lfloor k/t \rfloor > 0$ is

similar and omitted. □

CHAPTER III

COMPLETELY-DATA-DRIVEN SMOOTHING: THE UNIVARIATE CASE

3.1 Introduction

Consider one-dimensional functional estimation where we have samples (x_i, y_i), $i = 1, \cdots, n$ ($x_i \in \mathbb{R}$, $y_i \in \mathbb{R}$). To find the functional relationship depicted by $y_i = f(x_i) + \varepsilon_i$, a standard approach is to assume $f \in \mathscr{F}$, where $\mathscr{F}$ includes all functions with squared integrable second-order derivative, i.e., $\mathscr{F} = \left\{f : \int_{\mathbb{R}} |f^{(2)}(x)|^2 < \text{Constant}\right\}$, and solve

$$\min_{f \in \mathscr{F}} \sum_{i=1}^{n} (y_i - f(x_i))^2 + \lambda \int_{\mathbb{R}} |f^{(2)}(x)|^2 dx. \tag{3.1.1}$$

It is well known that the solution to the above is the cubic smoothing spline ([64]). Its asymptotic properties have been explicitly derived: (1) It achieves the optimal convergence rate ([59]) that is known in the general framework of nonparametric regression ([61]). (2) The hidden constant in the minimax convergence rate is optimal ([40]). There is an alternative way to derive an estimator of f. The derivation does not include an integration of the squared second-order derivative. The basic idea is to replace the penalty term in (3.1.1) by its local least squares estimator. More details follow in Section 3.2. It is interesting to find out whether the results of [59] and [40] still hold for the new estimator. This chapter establishes such results.

The remainder of this chapter is organized as follows. In Section 3.2, a completely-data-driven smoothing method is presented. The asymptotic properties of our data-driven method, which include optimal rate of convergence and asymptotic optimality, are presented in Section 3.3.

3.2 The data-driven method

We derive our method in this section. Formulation and regularization are revisited with needed additional details in Section 3.2.1. In Section 3.2.2, we derive a least-squares estimator of the integration of the second-order derivative. We then give the formula of our estimator in Section 3.2.3. Some basic properties of this estimator is presented as well. The tuning parameter λ is chosen by GCV as in smoothing splines ([64]).

3.2.1 Formulation

Recall that Ω is a bounded subset of the one-dimensional Euclidean space ($\mathbb{R}$). Assume the formulation in the introductory section. The penalty term in (3.1.1) can be estimated as follows:

$$\widetilde{\mathcal{J}}(f) = \sum_{i=1}^{n} \frac{x_{i+1} - x_{i-1}}{2} |f^{(2)}(x_i)|^2,$$

with sorted $\{x_1 < x_2 < \cdots < x_n\} \subset \Omega = [a, b] = [x_0, x_{n+1}]$. Therefore we come to optimize the following:

$$\min_{f \in \mathscr{F}} Q(f) = \frac{1}{n} \sum_{i=1}^{n} |y_i - f(x_i)|^2 + \lambda (\frac{1}{n} \sum_{i=1}^{n} |h_i|^2), \tag{3.2.1}$$

where $h_i = f^{(2)}(x_i)$.

In the following subsections, we will establish the path leading to the solution of (3.2.1).

3.2.2 Local estimate of second-order derivatives

The key step in our strategy to solve the model of (3.2.1) is utilizing the function values at x_i and its neighbors to estimate h_i, i.e., the second-order derivative $f^{(2)}(x_i)$. Let $\mathscr{B}_k(x_i) = \{x_{j_1}, \cdots, x_{j_k}\}$ be the k nearest neighbors of x_i. A local quadratic approximation of f is provided by the Taylor expansion

$$f(x_j) - f(x_i) \approx (x_j - x_i) f^{(1)}(x_i) + \frac{1}{2}(x_j - x_i)^2 f^{(2)}(x_i), \quad x_j \in \mathscr{B}_k(x_i), \tag{3.2.2}$$

where $f^{(1)}(x_i)$ and $f^{(2)}(x_i)$ are the first-order and second-order derivatives of f at x_i, respectively. For each x_i with its k nearest neighbors $\mathscr{B}_k(x_i) = \{x_{j_1}, \cdots, x_{j_k}\}$, we denote

$$\mathbf{L} = \begin{pmatrix} -1 & 1 & \cdots & \mathbf{0} \\ \vdots & \vdots & \ddots & \vdots \\ -1 & \mathbf{0} & \cdots & 1 \end{pmatrix} \in \mathbb{R}^{k\times(k+1)},$$

$\mathbf{f}_i = (f(x_i), f(x_{j_1}), \cdots, f(x_{j_k}))^T \in \mathbb{R}^{k+1}$, $\mathbf{p}_\ell = x_{j_\ell} - x_i$, $\mathbf{v}_\ell = \frac{1}{2}\mathbf{p}_\ell^2$, $\ell = 1, \cdots, k$. Then a matrix version of the local approximation is given by

$$\mathbf{L}\mathbf{f}_i \approx \mathbf{P}\mathbf{g}_i + \mathbf{V}\mathbf{h}_i = (\mathbf{P}, \mathbf{V}) \begin{pmatrix} g_i \\ h_i \end{pmatrix}, \tag{3.2.3}$$

where $\mathbf{P} = (p_1, \cdots, p_k)^T \in \mathbb{R}^k$, $\mathbf{V} = (v_1, \cdots, v_k)^T \in \mathbb{R}^k$ are constant vectors, and $g_i = f^{(1)}(x_i)$, $h_i = f^{(2)}(x_i)$ are unknowns.

We can then easily derive the least squares estimator of h_i in a closed form

$$\hat{h}_i = \mathbf{H}_i\mathbf{f}_i, \tag{3.2.4}$$

where $\mathbf{H}_i = \left(\mathbf{V}^T\mathbf{V} - \mathbf{V}^T\mathbf{P}(\mathbf{P}^T\mathbf{P})^{-1}\mathbf{P}^T\mathbf{V}\right)^{-1}\left(\mathbf{V}^T - \mathbf{V}^T\mathbf{P}(\mathbf{P}^T\mathbf{P})^{-1}\mathbf{P}^T\right)\mathbf{L} \in \mathbb{R}^{1\times(k+1)}$. Then the local estimate of squared second-order derivative is given by

$$|f^{(2)}(x_i)|^2 \approx |\hat{h}_i|^2 = \mathbf{f}_i\mathbf{H}_i^T\mathbf{H}_i\mathbf{f}_i. \tag{3.2.5}$$

3.2.3 Global solution to the approximation model

For each $i \in \{1, \cdots, n\}$, let $\mathbf{S}_i$ be a $(k+1)\times$ n matrix with 0 and 1 as its components and satisfy

$$(x_i, x_{j_1}, \cdots, x_{j_k}) = (x_1, x_2, \cdots, x_n)\mathbf{S}_i^T,$$

i.e., $\mathbf{S}_i$ selects the ith observation and its k nearest neighbors. It is evident to have $\mathbf{f}_i = \mathbf{S}_i\mathbf{f}$ with $\mathbf{f} = (f(x_1), \cdots, f(x_n))^T \in \mathbb{R}^n$. Then the objective function in (3.2.1) results in a quadratic form

$$\sum_{i=1}^{n} |y_i - f(x_i)|^2 + \lambda \sum_{i=1}^{n} |h_i|^2 = \|\mathbf{y} - \mathbf{f}\|^2 + \lambda(\mathbf{f}^T\mathbf{M}\mathbf{f}), \tag{3.2.6}$$

where $\mathbf{y} = (y_1, \cdots, y_n)^T \in \mathbb{R}^n$, $\mathbf{M} = \sum_{i=1}^{n} \mathbf{Q}_i^T \mathbf{Q}_i \in \mathbb{R}^{n\times n}$ and $\mathbf{Q}_i = \mathbf{H}_i \mathbf{S}_i \in \mathbb{R}^{1\times n}$. The approximation model eventually leads to solving linear systems

$$(\mathbf{I}_n + \lambda \mathbf{M})\hat{\mathbf{f}} = \mathbf{y}, \tag{3.2.7}$$

where $\hat{\mathbf{f}}$ is the global estimate of $\{f(x_i)\}_{i=1}^n$. Accordingly, the solution operator $\mathcal{A}_{n,\lambda} : \mathbb{R}^n \to \mathbb{R}^n$, is given for $\mathbf{y} \in \mathbb{R}^n$ by $\mathcal{A}_{n,\lambda}(\mathbf{y}) = \mathbf{A}_n(\lambda)\mathbf{y} = (\mathbf{I}_n + \lambda \mathbf{M})^{-1}\mathbf{y}$, which minimizes $\mathcal{Q}(f) = \frac{1}{n}\sum_{i=1}^n |y_i - f(X_i)|^2 + \lambda(\frac{1}{n}\sum_{i=1}^n |h_i|^2)$.

3.3 Theoretical properties

We discuss the asymptotic properties of the new estimator in this section. Specifically, we show that the new estimator enjoys all the nice properties presented in [59] for smoothing splines. In section 3.3.1, we study the behavior of the eigenvalues of the matrix $\mathbf{M}$ associated with our estimator. The optimal rate of convergence and sharp bound for the minimax risk are discussed in sections 3.3.2 and 3.3.3 respectively. All the derivations are based on equally spaced design.

3.3.1 Eigenvalues associated with equally spaced design

Suppose $\{x_i\}_{i=1}^n$ are equally spaced, i.e., $\delta = x_{i+1} - x_i$ for $1 \le i \le n-1$. Let $k = 2q \ll n$ and denote $\mathbf{f} = (f(x_1), \cdots, f(x_n))^T$. Then we have, for $q+1 \le i \le n-q$, $\mathbf{Q}_i$ in Section 3.2.3 has the form

$$\frac{1}{\gamma_q \delta^2}(0, \cdots, 0, q^2, (q-1)^2, \cdots, 1, -2\beta_q, 1, \cdots, (q-1)^2, q^2, 0, \cdots, 0) \in \mathbb{R}^n, \tag{3.3.1}$$

where $-2\beta_q$ is the ith component, $\beta_q = \sum_{\ell=1}^{q} \ell^2$ and $\gamma_q = \sum_{\ell=1}^{q} \ell^4$. The derivation of (3.3.1) is given in the appendix. It can be shown that $\mathbf{M}$ is a symmetric and banded matrix of bandwidth $2k+1$. After eliminating the first and last k rows and columns of $\mathbf{M}$, the remaining submatrix is Toeplitz. Let $\widetilde{\mathbf{M}}$ be a cyclic matrix that has the same dimension as matrix $\mathbf{M}$. In addition, after the aforementioned elimination, matrix $\mathbf{M}$ and $\widetilde{\mathbf{M}}$ share the

same central submatrix, denoted by $\mathbf{M}^{(1)} \in \mathbb{R}^{(n-2k)\times(n-2k)}$. It is known that the eigenvalues of $\widetilde{\mathbf{M}}$ can be computed in closed form. Hence we can combine this fact and the Cauchy interlacing theorem to give bounds on the eigenvalues of $\mathbf{M}$.

Denote $\tau_j = \frac{(j-\frac{1}{2})\pi}{n}$, $\tilde{\mathbf{u}}_j = (\sin(\tau_j), \sin(2\tau_j), \cdots, \sin(n\tau_j))^T$, $j = 1, \cdots, n$. It is not difficult to verify the following eigen-equations

$$\widetilde{\mathbf{M}}\tilde{\mathbf{u}}_j = \xi_j \tilde{\mathbf{u}}_j = \frac{4}{\gamma_q^2 \delta^4}\Big(\sum_{\ell=1}^{q} \ell^2(\cos \ell\tau_j - 1)\Big)^2 \tilde{\mathbf{u}}_j, \quad j = 1, \cdots, n. \tag{3.3.2}$$

Let $\mu_1 \le \cdots \le \mu_n$ and $\mu_1^{(1)} \le \cdots \le \mu_{n-2k}^{(1)}$ be the eigenvalues of $\mathbf{M}$ and $\mathbf{M}^{(1)}$, respectively. By twice applying the Cauchy interlacing theorem, we have, for $2k + 1 \le j \le n - 2k$,

$$\xi_{j-2k} \le \mu_{j-2k}^{(1)} \le \mu_j \le \mu_j^{(1)} \le \xi_{j+2k}. \tag{3.3.3}$$

Now we can bound the eigenvalues of $\mathbf{M}$ by using Jordan's inequality, which states that $\frac{2}{\pi}x \le \sin(x) \le x$ for any $x \in [0, \pi/2]$.

Lemma 3.3.1. *For any $\ell \in \mathbb{N}$ and $0 \le \tau \le \pi$, the cosine function has the following expansion*

$$\cos(\ell\tau) - 1 = (\cos\tau - 1)\Big[\ell^2 - O_+((\sin\frac{\tau}{2})^2)\Big], \tag{3.3.4}$$

where $O_+(\cdot)$ means that $0 \le \lim_{x\to 0} \frac{O_+(x)}{x} < +\infty$.

The proof just needs basic calculus and the details are thus omitted.

Lemma 3.3.2. *For $2k + 1 \le j \le n$, there exist constants $B_1, B_2 > 0$ such that*

$$B_1 j^4 \le \mu_j \le B_2 j^4. \tag{3.3.5}$$

Proof. Since $(\cos\tau - 1)^2 = \sin^4(\frac{\tau}{2})$, the conclusion in (3.3.5) immediately follows from Lemma 3.3.1, (3.3.2), (3.3.3) and Jordan's inequality. □

3.3.2 The optimal rate of convergence

For an appropriate $\lambda > 0$ and given $\mathbf{y} = (y_1, \cdots, y_n)^T$, let $\hat{\mathbf{f}}_{n,\lambda} = \mathcal{A}_{n,\lambda}(\mathbf{y}) = \mathbf{A}_n(\lambda)\mathbf{y}$ be the estimator from the approximation model. It is clear that $\mathcal{A}_{n,\lambda}$ is a linear operator. Therefore

$\mathcal{A}_{n,\lambda}(\mathbf{y}) = \mathcal{A}_{n,\lambda}(\mathbf{f}) + \mathcal{A}_{n,\lambda}(\varepsilon)$, where $\varepsilon = (\varepsilon_1, \cdots, \varepsilon_n)^T$ is the vector of noise. Thus the average mean squared error (AMSE) is given as

$$\begin{aligned} \text{AMSE}(\hat{\mathbf{f}}_{n,\lambda}) &= E\left[\tfrac{1}{n}(\hat{\mathbf{f}}_{n,\lambda} - \mathbf{f})^T(\hat{\mathbf{f}}_{n,\lambda} - \mathbf{f})\right] \\ &= \tfrac{1}{n}\|\mathcal{A}_{n,\lambda}(\mathbf{f}) - \mathbf{f}\|^2 + E\left[\tfrac{1}{n}\|\mathcal{A}_{n,\lambda}(\varepsilon)\|^2\right]. \end{aligned} \tag{3.3.6}$$

The cross term disappears because $E(\varepsilon) = \mathbf{0}$. We now study the decay rate of AMSE. Denote the eigen-decomposition of $\mathbf{M}$ as follows.

$$\mathbf{M} = \mathbf{U}\Lambda\mathbf{U}^T, \tag{3.3.7}$$

where $\Lambda = \text{diag}(\mu_1, \cdots, \mu_n)$ and $\mathbf{U} = (\mathbf{u}_1, \cdots, \mathbf{u}_n)$. The eigenvalues of $\mathbf{A}_n(\lambda) = (\mathbf{I}_n + \lambda\mathbf{M})^{-1}$ are then $a_j(\lambda) = \frac{1}{1+\lambda\mu_j}$, $j = 1, \cdots, n$. Therefore we have

$$\begin{aligned} \text{AMSE}(\hat{\mathbf{f}}_{n,\lambda}) &= \tfrac{1}{n}\mathbf{f}^T\left(\mathbf{A}_n(\lambda) - \mathbf{I}\right)^2\mathbf{f} + \tfrac{\sigma^2}{n}\text{tr}[\mathbf{A}(\lambda)^2] \\ &= \frac{1}{n}\sum_{j=1}^{n}\frac{\lambda^2\mu_j^2 b_j^2}{(1+\lambda\mu_j)^2} + \frac{\sigma^2}{n}\sum_{j=1}^{n}\frac{1}{(1+\lambda\mu_j)^2}, \end{aligned} \tag{3.3.8}$$

where $\mathbf{f} = (f(t_1), \cdots, f(t_n))^T$ and $\mathbf{b} = \mathbf{U}^T\mathbf{f} = (b_1, \cdots, b_n)^T$. The first term of $\text{AMSE}(\hat{\mathbf{f}}_{n,\lambda})$ in (3.3.8) is the average squared shrinkage bias and the second term is the average variance. We are now ready to show the convergence results for the new estimator.

We first present a result that is useful to prove the main theorem of the convergence rate.

Proposition 3.3.1. *If for $B_1, B_2 > 0$, we have $B_1 j^m \le \mu_j \le B_2 j^m$ for a constant $m > 0$ and $j = 1, 2, \ldots,$ then we have for $n > 0, \lambda > 0$,*

$$\sum_{j=1}^{n}\frac{1}{(1+\lambda\mu_j)^2} = O(\lambda^{-1/m}).$$

The proof is relatively straightforward and the details are omitted.

Theorem 3.3.1. *Suppose the functional class is the second-order Sobolev space W_2^2. The estimator $\hat{\mathbf{f}}_{n,\lambda}$ from the univariate CDS method converges to the true function $f(\cdot)$ with an optimal rate. Concretely, we have*

$$\text{AMSE}(\hat{\mathbf{f}}_{n,\lambda}) = O(n^{-4/5})$$

by choosing the smoothing parameter $\lambda = O(n^{-4/5})$.

Proof. For the first term of AMSE($\hat{\mathbf{f}}_{n,\lambda}$), we have

$$\frac{1}{n}\sum_{j=1}^{n}\frac{\lambda^2\mu_j^2 b_j^2}{(1+\lambda\mu_j)^2} = \frac{\lambda}{n}\sum_{j=1}^{n}\frac{\lambda\mu_j}{(1+\lambda\mu_j)^2}\mu_j b_j^2 \le \frac{\lambda}{4n}\sum_{j=1}^{n}\mu_j b_j^2 = \frac{\lambda}{4n}\mathbf{f}^T\mathbf{M}\mathbf{f} = O(\lambda), \tag{3.3.9}$$

since $\frac{1}{n}\mathbf{f}^T\mathbf{M}\mathbf{f} = O(\int_\Omega |f^{(2)}(t)|^2 \mathrm{d}t)$ is finite as $f \in W_2^2(\Omega)$.

For the second term, from Lemma 3.3.2 we know $\mu_j = O(j^4)$. Hence we have

$$\frac{\sigma^2}{n}\sum_{j=1}^{n}\frac{1}{(1+\lambda\mu_j)^2} = \frac{\sigma^2}{n}\sum_{j=1}^{n}\frac{1}{(1+\lambda O(j^4))^2} = O(\lambda^{-1/4}n^{-1}), \tag{3.3.10}$$

where the last equation is based on Proposition 3.3.1.

Furthermore, we can achieve the optimal decay rate of AMSE, i.e., AMSE($\hat{\mathbf{f}}_{n,\lambda}$) = $O(n^{-4/5})$, by choosing λ to be $O(n^{-4/5})$. □

3.3.3 Sharp bound for the minimax risk

Let $L_2 = L_2([0,1])$ be the Hilbert space of squared integrable functions on the unit interval, and $\|\cdot\|$ the usual norm therein. We denote $W_2^m = \{f \in L_2 \mid D^m f \in L_2\}$ the corresponding Sobolev space where $D^m f$ is the derivative of order m for $f \in L_2$, and let, for given $K > 0$,

$$W_2^{(m,K)} = \{f \in L_2 \mid \|D^m f\| \le K\}$$

be the nonparametric class of functions where f is from. $\mathscr{F}_n$ is defined to be the class of all estimators of f for given sample size n, i.e., measurable mappings $\hat{f} : \mathbb{R}^n \times [0,1] \to \mathbb{R}$. By applying the optimal filtering method of [45] to our approximation model, we can give an estimator $\tilde{f}$ which attains a sharp minimax risk bound presented in [40]. Based on the properties of eigenvalues $\{\mu_j\}$ (Lemma 3.3.3 and Lemma 3.3.4), we are now ready to prove a stronger result of optimal convergence rate.

For any real number x, denote by $\lfloor x \rfloor$ the largest integer less than or equal to x, and set $\tilde{n} = \lfloor n^{1/5}\log(n^{1/5})\rfloor$. Define a function $g(x) = (1-(\pi x)^m)_+(\mathrm{sgn}(x))_+$ and a number $\omega = \frac{1}{K}\int g(x)(1-g(x))\mathrm{d}x$. Further denote $\mathbf{G} = \sum_{j=1}^{n} g_j \mathbf{u}_j\mathbf{u}_j^T$, where $g_j = 1, j = 1, \cdots, \tilde{n}, g_j =$

$g(j\omega n^{-\frac{1}{2m+1}}), j = \tilde{n}+1, \cdots, n$, and $\{\mathbf{u}_j\}_{j=1}^n$ is the orthonormal eigensystem of $\mathbf{M}$ so that $\mathbf{M} = \sum_{j=1}^{n} \mu_j \mathbf{u}_j \mathbf{u}_j^T$. Let $\tilde{\mathbf{y}} = \mathbf{G}\mathbf{y}$, and define $\tilde{\mathbf{f}} = \mathcal{A}_{n,\lambda}(\tilde{\mathbf{y}})$ be the function values of estimator $\tilde{f}$ at the design points $\{x_j\}$.

Lemma 3.3.3. *There is a sequence $\{\eta_j\}$, not depending on n, $\lim_{j\to\infty} \eta_j = 1$, such that*

$$\sup_{1\le j\le \tilde{n}} \eta_j \mu_j (\pi j)^{-4} \le 1 + o(1), \quad n \to \infty. \tag{3.3.11}$$

Proof of the above lemma is in appendix.

Lemma 3.3.4. *There is a sequence $\{\eta_j\}$, not depending on n, $\lim_j \eta_j = 1$, such that*

$$\sup_{1\le j\le \tilde{n}} \eta_j \mu_j (\pi j)^{-4} \ge 1 + o(1), \quad n \to \infty. \tag{3.3.12}$$

The proof is similar to that for the previous lemma and is thus omitted.

Theorem 3.3.2. *Let $\mathscr{W} = W_2^{(2,K)}$ and $\Gamma(m,K) = (2m+1)^{\frac{1}{2m+1}} K^{\frac{1}{2m+1}} \left[\frac{m}{\pi(m+1)}\right]^{\frac{2m}{2m+1}}$. The normality of noise is assumed, i.e., $\{\varepsilon_i\}$ are independent random variables with normal distribution $N(0,\sigma^2)$. Then $\lim_{n\to\infty} \inf_{\hat{f}\in\mathscr{F}_n} \sup_{f\in\mathscr{W}} n^{\frac{4}{5}} E_f \|\hat{f} - f\|_T^2 = \sigma^2\Gamma(2,K)$, and the estimator $\tilde{f}$ attains this sharp bound.*

Proof. The conclusion follows immediately from Theorem 2.1 and Theorem 2.2 in [40] because the condition of Theorem 2.1 is fulfilled by Lemma 3.3.3 and the condition of Theorem 2.2 is implied by Lemma 3.3.4. □

3.4 Appendix

3.4.1 Derivation detail of (3.3.1)

Recall we assume that the sampled points $\{X_i\}_{i=1}^n$ are sorted and equally spaced, i.e., $\delta = X_{i+1} - X_i$ for $i = 1, \cdots, n-1$. Suppose $k = 2q$. At a sample point $X_i, q+1 \le i \le n-q$, we have

$$\mathscr{B}_k(X_i) = \{X_{i-q}, \cdots, X_{i-1}, X_{i+1}, \cdots, X_{i+q}\},$$

$$\mathbf{P} = (-q\delta, \cdots, -\delta, \delta, \cdots, q\delta)^T \in \mathbb{R}^{k\times 1},$$

$$\mathbf{V} = \frac{1}{2}(q^2\delta^2, \cdots, \delta^2, \delta^2, \cdots, q^2\delta^2)^T \in \mathbb{R}^{k\times 1}.$$

It is obvious that $\mathbf{P}^T\mathbf{V} = \mathbf{V}^T\mathbf{P} = 0$, $\mathbf{V}^T\mathbf{V} = \frac{1}{2}\gamma_q\delta^4$ and $\mathbf{V}^T\mathbf{L} = \frac{1}{2}\delta^2(-2\beta_q, q^2, \cdots, 1, 1, \cdots, q^2)$, with $\beta_q = \sum_{\ell=1}^{q} \ell^2$ and $\gamma_q = \sum_{\ell=1}^{q} \ell^4$. For (3.2.4), we have

$$\hat{h}_i = \mathbf{H}_i\mathbf{f}_i = (\mathbf{V}^T\mathbf{V})^{-1}\mathbf{V}^T\mathbf{L}\mathbf{f}_i = \frac{1}{\gamma_q\delta^2}(-2\beta_q, q^2, \cdots, 1, 1, \cdots, q^2)\mathbf{f}_i$$

where $\mathbf{f}_i = (f(X_i), f(X_{i-q}), \cdots, f(X_{i-1}), f(X_{i+1}), \cdots, f(X_{i+q}))^T \in \mathbb{R}^{(k+1)\times 1}$. Finally, we get

$$\mathbf{Q}_i = \mathbf{H}_i\mathbf{S}_i = \frac{1}{\gamma_q\delta^2}(\cdots, 0, q^2, (q-1)^2, \cdots, 1, -2\beta_q, 1, \cdots, (q-1)^2, q^2, 0, \cdots).$$

3.4.2 Proof of Lemma 3.3.1

Proof. Here we use the mathematical induction to prove (3.3.4) together with an additional statement

$$\sin\tau\sin(\ell\tau) = (1-\cos\tau)\left[2\ell - O_+((\sin\frac{\tau}{2})^2)\right]. \tag{3.4.1}$$

In the basis step, (3.3.4) is obviously true for $\ell = 1$, and (3.4.1) holds for $\ell = 1$ since

$$\sin\tau\sin\tau = 1 - (\cos\tau)^2 = (1-\cos\tau)(1+\cos\tau) = (1-\cos\tau)(2 - 2(\sin\frac{\tau}{2})^2).$$

In the inductive step, using the induction hypothesis that (3.3.4) and (3.4.1) hold for ℓ, we have

$$\begin{aligned}
\cos(\ell+1)\tau - 1 &= \cos\tau\cos\ell\tau - \sin\tau\sin\ell\tau - 1 \\
&= (\cos\tau - 1) + \cos\tau(\cos\ell\tau - 1) - \sin\tau\sin\ell\tau \\
&= (\cos\tau - 1)\left[1 + \cos\tau(\ell^2 - O_+((\sin\tfrac{\tau}{2})^2)) + 2\ell - O_+((\sin\tfrac{\tau}{2})^2)\right] \\
&= (\cos\tau - 1)\left[1 + (1 - 2(\sin\tfrac{\tau}{2})^2)(\ell^2 - O_+((\sin\tfrac{\tau}{2})^2)) + 2\ell - O_+((\sin\tfrac{\tau}{2})^2)\right] \\
&= (\cos\tau - 1)\left[(\ell+1)^2 - O_+((\sin\tfrac{\tau}{2})^2)\right],
\end{aligned}$$

$$\begin{aligned}
\sin\tau\sin(\ell+1)\tau &= \sin\tau(\sin\tau\cos\ell\tau + \cos\tau\sin\ell\tau) \\
&= (1 - (\cos\tau)^2) + (\sin\tau)^2(\cos\ell\tau - 1) + \cos\tau(\sin\tau\sin\ell\tau) \\
&= (1-\cos\tau)\left[1 + \cos\tau - (\sin\tau)^2(\ell^2 - O_+((\sin\tfrac{\tau}{2})^2)) + \cos\tau(2\ell - O_+((\sin\tfrac{\tau}{2})^2))\right] \\
&= (1-\cos\tau)\left[2 - 2(\sin\tfrac{\tau}{2})^2 - O_+((\sin\tfrac{\tau}{2})^2) + (1 - 2(\sin\tfrac{\tau}{2})^2)(2\ell - O_+((\sin\tfrac{\tau}{2})^2))\right] \\
&= (1-\cos\tau)\left[2(\ell+1) - O_+((\sin\tfrac{\tau}{2})^2)\right],
\end{aligned}$$

thereby showing that indeed (3.3.4) and (3.4.1) hold for $\ell+1$. This concludes the proof. □

3.4.3 Proof of Proposition 3.3.1

Proof. First we have

$$\sum_{j=1}^{n} \frac{1}{(1+\lambda B_2 j^m)^2} \le \sum_{j=1}^{n} \frac{1}{(1+\lambda \mu_j)^2} \le \sum_{j=1}^{n} \frac{1}{(1+\lambda B_1 j^m)^2}.$$

For $i = 1, 2$, we have

$$\begin{aligned}
\sum_{j=1}^{n} \frac{1}{(1+\lambda B_i j^m)^2} &\ge \int_1^{n+1} \frac{1}{(1+\lambda B_i x^m)^2} dx \\
&= \frac{1}{m} \int_{\lambda B_i}^{\lambda B_i (n+1)^m} \frac{y^{-\frac{m-1}{m}}}{(1+y)^2} dy \cdot (\lambda B_i)^{-1/m} \\
&\to m^{-1} \left(\int_{\lambda B_i}^{\infty} \frac{y^{-(m-1)/m}}{(1+y)^2} dy \right) B_i^{-1/m} \cdot \lambda^{-1/m} \\
&= O(\lambda^{-1/m}),
\end{aligned}$$

where the second equation reflects the change of variable ($y = \lambda B_i x^m$), and "$\to$" corresponds to "$n \to \infty$." Similarly, with the same change of variable, we also have

$$\begin{aligned}
\sum_{j=1}^{n} \frac{1}{(1+\lambda B_i j^m)^2} &\le \int_0^{n} \frac{dx}{(1+\lambda B_i x^m)^2} \\
&= \frac{1}{m} \int_{0}^{\lambda B_i n^m} \frac{y^{-\frac{m-1}{m}}}{(1+y)^2} dy \cdot (\lambda B_i)^{-1/m} \\
&\to m^{-1} \left(\int_{\lambda B_i}^{\infty} \frac{y^{-(m-1)/m}}{(1+y)^2} dy \right) B_i^{-1/m} \cdot \lambda^{-1/m} \\
&= O(\lambda^{-1/m}).
\end{aligned}$$

□

3.4.4 Proof of Lemma 3.3.3

Proof. Since $\mu_j \le \xi_{j+2k} = \frac{4}{\gamma_q^2\delta^4}\left[\sum_{\ell=1}^{q}\ell^2(\cos\ell\tau_{j+2k}-1)\right]^2$, we can apply Lemma 3.3.1 to obtain

$$\begin{aligned}\mu_j(\pi j)^{-4} &\le \frac{4(\cos\tau_{j+2k}-1)^2}{\gamma_q^2\delta^4(\pi j)^4}\left[\sum_{\ell=1}^{q}\ell^2(\ell^2-O_+((\sin\tfrac{\tau_{j+2k}}{2})^2))\right]^2\\ &= \frac{16\sin^4(\frac{\tau_{j+2k}}{2})}{\delta^4(\pi j)^4}\cdot\frac{\left[\sum_{\ell=1}^{q}\ell^4-O_+((\sin\frac{\tau_{j+2k}}{2})^2)\right]^2}{\gamma_q^2} \qquad (3.4.2)\\ &\le \frac{16(\frac{\tau_{j+2k}}{2})^4}{\delta^4(\pi j)^4} = \left(1+\tfrac{2k-1/2}{j}\right)^4,\end{aligned}$$

which implies

$$\sup_{1\le j\le\tilde{n}}\eta_j\mu_j(\pi j)^{-4}\le 1+o(1),\quad n\to\infty,$$

with setting $\eta_j=(\frac{j}{j+2k-1/2})^4$. □

3.4.5 Proof of Lemma 3.3.4

Proof. According to Lemma 3.3.1, we have for $2k+1\le j\le\tilde{n}$,

$$\begin{aligned}\mu_j(\pi j)^{-4} &\ge \frac{16\sin^4(\frac{\tau_{j-2k}}{2})}{\delta^4(\pi j)^4}\cdot\frac{\left[\sum_{\ell=1}^{q}\ell^4-O_+((\sin\frac{\tau_{j-2k}}{2})^2)\right]^2}{\gamma_q^2}\\ &\ge \frac{16\left(\frac{\tau_{j-2k}}{2}-(\frac{\tau_{j-2k}}{2})^3/6\right)^4}{\delta^4(\pi j)^4}\cdot\frac{\left[\sum_{\ell=1}^{q}\ell^4-O_+((\frac{\tau_{j-2k}}{2})^2)\right]^2}{\gamma_q^2} \qquad (3.4.3)\\ &\ge \frac{\tau_{j-2k}^4-\tau_{j-2k}^6/6}{\delta^4(\pi j)^4}\left(1-O_+((\tfrac{\tau_{j-2k}}{2})^2)\right)^2\\ &\ge \left((1-\tfrac{2k+1/2}{j})^4-\tfrac{\pi^2}{6}(\tfrac{\tilde{n}}{n})^2\right)\left(1-O_+((\tfrac{\tilde{n}}{n})^2)\right).\end{aligned}$$

Further letting $\eta_j=(\frac{j}{j-2k-1/2})^4$, we have

$$\sup_{1\le j\le\tilde{n}}\eta_j\mu_j(\pi j)^{-4}\ge 1+o(1),\quad n\to\infty$$

since $\lim_{n\to\infty}\frac{\tilde{n}}{n}=0$. □

CHAPTER IV

COMPLETELY-DATA-DRIVEN SMOOTHING: THE MULTIVARIATE CASE

4.1 Introduction

Consider the smoothing or functional estimation problem where we have observations y_i at inputs X_i, $i = 1, \ldots, n$ and the observations are assumed to satisfy

$$y_i = f(X_i) + \varepsilon_i, \tag{4.1.1}$$

where ε_i are i.i.d random errors. Suppose the domain of the inputs is $\Omega \in \mathbb{R}^d$. In this chapter, we consider the case when $d > 1$, i.e., the multivariate input case. As [52] mentioned, four classes of smoothing methods which have received much attention in the literature are kernel smoothing, wavelet-based methods, kriging and spline smoothing. See [66], [8], [24], [41] for the details. This chapter is concerned with the class of spline smoothers. For smoothing splines, one typically assumes the underlying function comes from an mth-order Sobolev space and construct an objective function that is a trade-off between the fidelity to the observed data and the smoothness of the estimated function. The solution to the constructed objective function is called the D^m-smoothing splines. For more details, see [64].

In most smoothing contexts, the domain Ω is assumed to be bounded with some regularity conditions such as connected and having a convex boundary. In such a case, any two points in the domain can be connected by a straight line segment lying entirely within the domain and Euclidean distance is a natural measure of closeness. However, in some cases, Euclidean distance can become an inappropriate measure of closeness between points. [52] considered smoothing the estimated per capita income data over the island of Montreal. It used three specific locations to illustrate the inappropriateness of Euclidean distance as a

measure of closeness. The reason that Euclidean distance becomes inappropriate is that the domain of interest is not *regular*, with holes inside and complicated boundary. Unfortunately, popular smoothing methods, such as thin plate spline smoothing, kriging and kernel smoothing, as they are usually defined, do not perform well for smoothing over irregular domains. To be more specific, as pointed out in [52], the thin plate spline suffers from problems with concave boundaries and holes in the domain. This is because its roughness penalty is integrated over the whole real plane and thus produces a function which is smooth on $\mathbb{R}^2$ rather than just on Ω. [61], based on the earlier work by [18], defined a technique for approximating a thin plate spline with the roughness penalty integrated over a simply connected (no holes) finite domain. The technique performed well in subregions of the domains not too close to the boundary. However, [18] reported that the approximation is relatively poor close to the boundary and suggested remedying the problem by estimating the solution over a somewhat larger domain $\Omega' \supset \Omega$. [52] proposed a new penalty term and tried to circumvent the boundary issue. His approach is shown to be a huge improvement on conventional smoothers, such as thin plate splines, over difficult domains. But the method involves quite a complex computational strategy. To arrive at the computational approach, [52] made strong assumptions on the boundary, i.e., at the boundary, the gradient of the estimated function is 0 along normals to the boundary. This implies that contours of the estimated function must meet the boundary at right angles, which is a strong assumption. [72] developed a new technique called soap film smoothing to tackle the smoothing problem over difficult domains. With reasonable assumptions on the boundary, soap film smoothing is demonstrated to perform better than thin plate splines and the method in [52]. However, one has to define the boundary rigorously in order to use soap film. Sometimes, this can be very inconvenient in practice. Moreover, theoretical properties of soap such as consistency, convergence rates, are unknown.

In this chapter, we propose a new method, called completely-data-driven smoothing, for smoothing over difficult domains when there is a relatively large sample size. The

idea is to replace the penalty term in the objective function of smoothing splines with its estimate. The estimation is achieved by deploying Taylor expansion and local least squares method. More details follow in Section 4.2. Large samples are more frequently met these days, especially in spatial problems, due to the large region involved and new technology to collect data. We will show that the solution to our reformulated problem has a close form which can be easily implemented in practice. We summarize the merits of our method in the following aspects:

1. When the domain of inputs is irregular, the smoothing splines solution can be challenging or even impossible to compute. By contrast, for any domain, our method has a close form which can be easily implemented in practice. Moreover, with a relatively large sample size, it seems that our method can circumvent the inappropriateness of Euclidean distance as a measure of closeness. This is because we only use the information of its nearest neighbors to estimate the penalty at a specific point. With a relatively large sample size and a relatively small number of nearest neighbors, the bias introduced by using Euclidean distance is reduced. This can be the reason why our method performs better than thin plate splines for difficult domains. See the numerical results in Section 4.4.

2. We use local least squares to estimate penalty at observed points. Thus the matrix in our estimator is sparse, which reduces the computation complexity significantly. A more detailed analysis is in Section 4.4.

3. With the same regularity assumptions on the boundary for the D^m-smoothing splines as in [63] and some regularity assumptions on the underlying functional class, we can prove that our method achieves the optimal convergence rate in L_2 sense.

The remainder of the chapter is organized as follows. In Section 4.2, a completely-data-driven method is proposed for functional estimation. In Section 4.3, we prove the

asymptotic properties of our data-driven method. In Section 4.4, the computational complexity of our algorithm is analyzed and numerical comparison is conducted to demonstrate the proposed method's performance. Finally we conclude the chapter with a brief review of problems for future research in Section 4.5.

4.2 The data-driven method

We derive our method in this section. We first give some notations and problem formulation in Section 4.2.1. A local least squares estimator based on Taylor expansion is then proposed in Section 4.2.2. We then give the close form of our estimator in Section 4.2.3. Section 4.2.4 briefly describes the adoption of GCV (generalized cross validation) to choose the associated regularization parameter.

4.2.1 Notations and problem formulation

We first review the formulation of D^m-smoothing splines. Let $\Omega \subset \mathbb{R}^d$ be the domain of the inputs. Let $H^m(\Omega)$, $m > 1$, be the Sobolev space

$$H^m(\Omega) = \{f \in \mathscr{D}^{'}(\Omega) | \int_{\Omega} \sum_{|\alpha| \leq m} ||D^{\alpha} f||^2 < +\infty\}, \tag{4.2.1}$$

where $\mathscr{D}^{'}(\Omega)$ is the space of Schwartz distributions and

$$D^{\alpha} f = \frac{\partial^{|\alpha|}}{\partial x_1^{\alpha_1} \cdots \partial x_d^{\alpha_d}} f \tag{4.2.2}$$

is the usual multi-index notation for partial derivatives.

For $m > d/2$, let f be an element of $H^m(\Omega)$ and $y_1, \cdots, y_n$ satisfy

$$y_i = f(X_i) + \varepsilon_i, \ i = 1, \cdots, n, \tag{4.2.3}$$

where y_i are the observed function values at X_i, $i = 1, \cdots, n$ and ε_i, $i = 1, \cdots, n$ are i.i.d random variables with zero mean and variance σ^2.

To approximate f, it has been proposed to use the D^m-smoothing splines ([64]) defined as the unique solution to the variational problem

$$\min_{f \in D^{-m} L^2(\mathbb{R}^d)} \frac{1}{n} \sum_{i=1}^{n} (y_i - f(X_i))^2 + \lambda \mathcal{J}(f), \tag{4.2.4}$$

where

$$\mathcal{J}(f) = \sum_{i_1,\cdots,i_m=1}^{d} \int_{\mathbb{R}^d} |\frac{\partial^m f(X)}{\partial x_{i_1} \cdots \partial x_{i_m}}|^2 \mathrm{d}X, \tag{4.2.5}$$

and

$$D^{-m}L^2(\mathbb{R}^d) = \{f \in \mathscr{D}'(\mathbb{R}^d) | D^\alpha f \in L^2(\mathbb{R}^d), |\alpha| = m\}. \tag{4.2.6}$$

Denote $T = \{X_1, \cdots, X_n\}$. A finite collection of points $S \subset \mathbb{R}^d$ is called $\mathscr{P}^d_{m-1}$-unisolvent if any $p(X) \in \mathscr{P}^d_{m-1}$ is uniquely determined by its values on S, where $\mathscr{P}^d_{m-1}$ is the set of polynomials defined on $\mathbb{R}^d$ of total degree less than or equal to $m-1$. It has been proved by [17] that (4.2.4) has a unique solution provided T contains a $\mathscr{P}^d_{m-1}$-unisolvent set.

The smoothing splines are widely studied in statistics and have many applications in regression problems. Generally it is costly to compute the D^m-smoothing splines solution to (4.2.4) when $d > 1$. Moreover, the spline fitting problems can have condition numbers in excess of 10^9, which can potentially cause numerical instabilities (see [69]). Also, as mentioned in the introduction, this formulation does not work well for irregular regions.

To resolve the aforementioned issues, we propose to replace $\mathcal{J}(f)$ with its approximate:

$$\widetilde{\mathcal{J}}(f) = \sum_{i=1}^{n} \|\mathbf{h}_i\|^2 |\triangle_i|, \tag{4.2.7}$$

where $\triangle = \{\triangle_i : X_i \in \triangle_i\}_{i=1}^n$ is a partition of domain Ω, $|\triangle_i|$ denotes the volume of $\triangle_i$, and $\|\mathbf{h}_i\|^2 = \sum_{i_1,\cdots,i_m=1}^{d} |\frac{\partial^m f(X_i)}{\partial x_{i_1} \cdots \partial x_{i_m}}|^2$.

Typically, $|\triangle_i| = \frac{\mathrm{Vol}(\Omega)}{n}$, $i = 1, \cdots, n$, are specified in (4.2.7) for uniform sampling. Therefore we come to an optimization of the following model:

$$\min_{f \in D^{-m}(L_2(\mathbb{R}^d))} Q(f) = \frac{1}{n}\sum_{i=1}^{n}(y_i - f(X_i))^2 + \lambda(\frac{1}{n}\sum_{i=1}^{n}\|\mathbf{h}_i\|^2). \tag{4.2.8}$$

From now on, we name the model the CDS model. In the following subsections, we derive the solution to (4.2.8).

4.2.2 Local estimate of the penalty term

The key idea to derive the solution to (4.2.8) is to utilize the function values at X_i and its neighbors to estimate $\mathbf{h}_i$. Let $\mathscr{B}_k(X_i) = \{X_{j_1}, \cdots, X_{j_k}\}$ be the k nearest neighbors of X_i. Let

$\alpha = (\alpha_1, \cdots, \alpha_d) \in \mathbb{Z}_+^d$ be the multi-index defined as before. Let $B_\alpha(X) = \frac{x_1^{\alpha_1} \cdots x_d^{\alpha_d}}{\alpha_1! \cdots \alpha_d!}$. Given X_i, the Taylor expansion of f at its neighboring points can be written as follows:

$$f(X_j) - f(X_i) \approx \sum_{|\alpha|=1}^{m} (D^\alpha f(X_i))^T B^\alpha (X_j - X_i), \quad X_j \in \mathscr{B}_k(X_i). \tag{4.2.9}$$

For each X_i with its k nearest neighbors $\mathscr{B}_k(X_i) = \{X_{j_1}, \cdots, X_{j_k}\}$, we denote

$$\mathbf{L} = \begin{pmatrix} -1 & 1 & \cdots & \mathbf{0} \\ \vdots & \vdots & \ddots & \vdots \\ -1 & \mathbf{0} & \cdots & 1 \end{pmatrix} \in \mathbb{R}^{k \times (k+1)},$$

$\mathbf{f}_i = (f(X_i), f(X_{j_1}), \cdots, f(X_{j_k}))^T \in \mathbb{R}^{k+1}$, $\mathbf{p}_\ell = ((B^1(X_{j_\ell} - X_i))^T, \cdots, (B^{m-1}(X_{j_\ell} - X_i))^T)^T$, $\mathbf{v}_\ell = B^m(X_{j_\ell} - X_i)$, $\ell = 1, \cdots, k$, where $B^j(X)$ is the vector of all $B^\alpha(X)$'s such that $|\alpha| = j$, $j = 1, \cdots, m$. Then a matrix version of the local approximation is given by

$$\mathbf{L}\mathbf{f}_i \approx \mathbf{P}\mathbf{g}_i + \mathbf{V}\mathbf{h}_i = (\mathbf{P}, \mathbf{V}) \begin{pmatrix} \mathbf{g}_i \\ \mathbf{h}_i \end{pmatrix}, \tag{4.2.10}$$

where $\mathbf{P} = (\mathbf{p}_1, \cdots, \mathbf{p}_k)^T$, $\mathbf{V} = (\mathbf{v}_1, \cdots, \mathbf{v}_k)^T$ are constant vectors, and $\mathbf{g}_i = ((D^1 f(X_i))^T, (D^{m-1} f(X_i))^T)^T$, $\mathbf{h}_i = D^m f(X_i)$ are unknowns. From (4.2.10), we have

$$\begin{pmatrix} \mathbf{P}^T\mathbf{P} & \mathbf{P}^T\mathbf{V} \\ \mathbf{V}^T\mathbf{P} & \mathbf{V}^T\mathbf{V} \end{pmatrix} \begin{pmatrix} \mathbf{g}_i \\ \mathbf{h}_i \end{pmatrix} = \begin{pmatrix} \mathbf{P}^T \\ \mathbf{V}^T \end{pmatrix} \mathbf{L}\mathbf{f}_i.$$

A necessary condition for this linear system to have a unique solution is $k > M$, where $M = \sum_{\ell=1}^{m} \frac{(d-1+\ell)!}{(d-1)!\ell!}$. In addition, if $\mathscr{B}_k(X_i)$ contains a $\mathscr{P}_{m-1}^d$-unisolvent subset, then $k > M$ is also sufficient. Since we can choose k in the numerical calculation and we assume a relatively large number of observations, we assume the solution to the above linear system is unique without further discussion. From the above linear system, we have

$$\begin{pmatrix} \mathbf{P}^T\mathbf{P} & \mathbf{P}^T\mathbf{V} \\ 0 & \mathbf{V}^T\mathbf{V} - \mathbf{V}^T\mathbf{P}(\mathbf{P}^T\mathbf{P})^{-1}\mathbf{P}^T\mathbf{V} \end{pmatrix} \begin{pmatrix} \mathbf{g}_i \\ \mathbf{h}_i \end{pmatrix} = \begin{pmatrix} \mathbf{P}^T \\ \mathbf{V}^T - \mathbf{V}^T\mathbf{P}(\mathbf{P}^T\mathbf{P})^{-1}\mathbf{P}^T \end{pmatrix} \mathbf{L}\mathbf{f}_i,$$

and achieve a least squares estimator of $\mathbf{h}_i$ in the following form

$$\hat{\mathbf{h}}_i = \mathbf{H}_i \mathbf{f}_i, \tag{4.2.11}$$

where $\mathbf{H}_i = \left(\mathbf{V}^T\mathbf{V} - \mathbf{V}^T\mathbf{P}(\mathbf{P}^T\mathbf{P})^{-1}\mathbf{P}^T\mathbf{V}\right)^{-1}\left(\mathbf{V}^T - \mathbf{V}^T\mathbf{P}(\mathbf{P}^T\mathbf{P})^{-1}\mathbf{P}^T\right)\mathbf{L}$. Then the local estimate of $\|D^m f(X_i)\|^2$ is given by

$$\|D^m f(X_i)\|^2 \approx \|\hat{\mathbf{h}}_i\|^2 = \mathbf{f}_i^T\mathbf{H}_i^T\mathbf{H}_i\mathbf{f}_i. \tag{4.2.12}$$

4.2.3 Global solution to the CDS model

For each $i \in \{1, \cdots, n\}$, let $\mathbf{S}_i$ be a $\mathbb{R}^{(k+1)\times n}$ matrix with 0 and 1 as its components and satisfy

$$(X_i, X_{j_1}, \cdots, X_{j_k}) = (X_1, X_2, \cdots, X_n)\mathbf{S}_i^T,$$

i.e., $\mathbf{S}_i$ selects the ith observation and its k nearest neighbors. We thus have $\mathbf{f}_i = \mathbf{S}_i\mathbf{f}$, where $\mathbf{f} = (f(X_1), \cdots, f(X_n))^T \in \mathbb{R}^n$. Then the objective function of optimization in (4.2.8) results in a quadratic form

$$\sum_{i=1}^{n}(y_i - f(X_i))^2 + \lambda\sum_{i=1}^{n}\|\mathbf{h}_i\|^2 = \|\mathbf{y} - \mathbf{f}\|^2 + \lambda(\mathbf{f}^T\mathbf{M}\mathbf{f}), \tag{4.2.13}$$

where $\mathbf{y} = (y_1, \cdots, y_n)^T \in \mathbb{R}^n$, $\mathbf{M} = \sum_{i=1}^{n}\mathbf{Q}_i^T\mathbf{Q}_i \in \mathbb{R}^{n\times n}$ and $\mathbf{Q}_i = \mathbf{H}_i\mathbf{S}_i$. The CDS model eventually leads to solving linear systems

$$(\mathbf{I}_n + \lambda\mathbf{M})\hat{\mathbf{f}} = \mathbf{y}, \tag{4.2.14}$$

where $\hat{\mathbf{f}}$ is the global estimate of function evaluations $\{f(X_i)\}_{i=1}^n$. Accordingly, the solution operator $\mathcal{A}_{n,\lambda} : \mathbb{R}^n \to \mathbb{R}^n$, is given for $\mathbf{y} \in \mathbb{R}^n$ by $\mathcal{A}_{n,\lambda}(\mathbf{y}) = \mathbf{A}_n(\lambda)\mathbf{y} = (\mathbf{I}_n + \lambda\mathbf{M})^{-1}\mathbf{y}$, which minimizes $Q(f)$ in (4.2.8).

4.2.4 Choice of the penalty parameter λ

We adopt the generalized cross validation (GCV) to determine the penalty parameter λ. The generalized cross validation function is defined as

$$\mathrm{GCV}_n(\lambda) = \frac{1}{n}\sum_{i=1}^{n}\left(\frac{y_i - \hat{f}_{n,\lambda}(X_i)}{1 - \frac{1}{n}\mathrm{tr}[\mathbf{A}_n(\lambda)]}\right)^2, \tag{4.2.15}$$

where $\mathbf{A}_n(\lambda) = (\mathbf{I}_n + \lambda\mathbf{M})^{-1}$. The optimal value of the penalty parameter λ is chosen by minimizing the above GCV function, i.e.,

$$\hat{\lambda}_G = \arg\min_{\lambda>0} \mathrm{GCV}_n(\lambda). \tag{4.2.16}$$

The justification is relatively straightforward and can be found in many places, e.g., [64].

4.3 Theoretical properties

We establish the theoretical properties of CDS in this section. In Section 4.3.1, We briefly review the *m*th-order Sobolev space of generalized functions and give some definition and notation needed later. In Section 4.3.2, we derive the convergence rates of CDS under some regularity conditions. All the technical proofs are left in appendix.

4.3.1 A brief review of Sobolev space

Let Ω, α, T, $D^\alpha f$, and $D^{-m}L_2(\mathbb{R}^d)$ be defined as in Section 4.2.1. Similar to the definition of $D^{-m}L_2(\mathbb{R}^d)$, define $D^{-m}L_2(\Omega)$ as

$$D^{-m}L^2(\Omega) = \{f \in \mathscr{D}'(\Omega) | D^\alpha f \in L^2(\Omega), |\alpha| = m\}. \tag{4.3.1}$$

For $0 \le \ell \le m$, a semi-inner-product in $D^{-m}L_2(\Omega)$ is defined by

$$(f, g)_{\Omega,\ell} = \int_\Omega \sum_{|\alpha|=\ell} \frac{\ell!}{\alpha!} (D^\alpha f)(D^\alpha g) \mathrm{d}X, \tag{4.3.2}$$

which gives rise to the related semi-norm

$$|f|^2_{\Omega,\ell} = \int_\Omega \sum_{|\alpha|=\ell} \frac{\ell!}{\alpha!} |D^\alpha f|^2 \mathrm{d}X. \tag{4.3.3}$$

Next, define a discrete version of the aforementioned semi-norm as

$$|f|^2_{T,\ell} = \frac{1}{n} \sum_{i=1}^n \sum_{|\alpha|=\ell} \frac{\ell!}{\alpha!} |D^\alpha f(X_i)|^2. \tag{4.3.4}$$

Specially, $|f|^2_{\Omega,0} = \int_\Omega f(X)^2 \mathrm{d}X$ and $|f|^2_{T,0} = \frac{1}{n} \sum_{i=1}^n f(X_i)^2$.

4.3.2 Convergence rate of CDS

Before proving the main results, we give a list of regularity assumptions as follows.

1. The input domain Ω is an open bounded Lipschitz domain satisfying the uniform cone condition. See [63] for detailed definition.

2. Given T, there exist constants $\underline{B}$ and $\overline{B}$ (depending on T) such that $\underline{B}|f|^2_{\Omega,m} \leq |f|^2_{T,m} \leq \overline{B}|f|^2_{\Omega,m}$ for any function f in $D^{-m}L_2(\Omega)$.

3. $T = \{X_i\}_{i=1}^n$ satisfies the following quasi-uniform assumption: there exists a constant $B_0 > 0$ such that

$$\frac{\delta_{\max}}{\delta_{\min}} \leq B_0, \tag{4.3.5}$$

   where $\delta_{\max} = \sup_{X\in\Omega} \inf_{X_i\in T} \|X - X_i\|$, and $\delta_{\min} = \min_{j\neq i} \|X_j - X_i\|$.

4. The underlying function f is from class $C^{m,1}(\Omega)$, i.e., $f \in C^m(\Omega)$ and the partial derivatives $D^\alpha f$ of order $|\alpha| = m$ are Lipschitz continuous in Ω.

5. For any X_i in T, $\mathscr{B}_k(X_i)$ is $\mathscr{P}_m$-unisolvent.

We now give a remark on the assumptions.

Remark 2. *Assumptions 1 and 3 are standard regularity assumptions in the existing literature. Suppose $T = \{X_i\}_{i=1}^n$ is an equidistributed sequence in the region Ω. From law of large numbers, we have*

$$\lim_{n\to\infty} |f|^2_{T,m} = \frac{1}{\mathrm{Vol}(\Omega)}|f|^2_{\Omega,m}.$$

Since Ω is bounded, $\mathrm{Vol}(\Omega)$ is also bounded. Thus we have Assumption 2 is satisfied with probability one as the sample size goes to infinity. Assumption 4 is a stronger assumption than those in the existing literature on the underlying function f. The validity of local Taylor expansion approximation in CDS is based on Assumption 4.

Given T, we now provide a very useful representation of $|f|^2_{T,m}$.

Proposition 4.3.1. *For any function f in $D^{-m}L_2(\Omega)$, there exists a matrix $\mathbf{E}_{T,m}$ (depending on T and m) such that*

$$|f|^2_{T,m} = \min_{\phi \in D^{-m}L_2(\Omega), \phi(X_i)=f(X_i)} |\phi|^2_{T,m} = \frac{1}{n}\mathbf{f}^T \mathbf{E}_{T,m}\mathbf{f}, \tag{4.3.6}$$

where $\mathbf{f} = (f(X_1), \cdots, f(X_n))$.

The proof of the above proposition can be found in standard functional analysis textbooks and the details are thus omitted.

Next we present a result on the largest eigenvalue of $\mathbf{E}_{T,m}$.

Lemma 4.3.1. *If Ω is a bounded domain in $\mathbb{R}^d$ and e_n is the largest eigenvalue of matrix $\mathbf{E}_{T,m}$, then $n\delta^d_{\max}$ and $\delta^{2m}_{\max} e_n$ are both bounded from above, where $\delta_{\max}$ is defined in Assumption 3.*

We are now ready to exhibit the Rayleigh quotient inequalities connecting Sobolev semi-norms and their discretized version.

Lemma 4.3.2. *Let Ω satisfy Assumption 1 and $f \neq 0$ satisfy Assumption 2. Then there exist constants $C_1 > 0$ (depending on $d, m, \Omega, B_0, \underline{B}$) and $\delta_0 > 0$, such that if $\delta_{\max} \leq \delta_0$ we have*

$$\frac{|f|^2_{T,m}}{|f|^2_{T,0}} \geq \frac{|f|^2_{\Omega,m}}{C_1(|f|^2_{\Omega,0} + \delta^{2m}_{\max}|f|^2_{\Omega,m})}. \tag{4.3.7}$$

Lemma 4.3.3. *Assuming the same conditions as in Lemma 4.3.2, there exist constants $C_2 > 0$ (depending only on $d, m, \Omega, B_0, \underline{B}, \overline{B}$) and $\delta_0 > 0$, such that if $\delta_{\max} \leq \delta_0$ we have*

$$\frac{|f|^2_{\Omega,m}}{|f|^2_{\Omega,0}} \geq \frac{|f|^2_{T,m}}{C_2(|f|^2_{T,0} + \delta^{2m}_{\max}|f|^2_{T,m})}, \tag{4.3.8}$$

for any $0 \neq f$.

Lemmas 4.3.2 and 4.3.3 build a connection between the continuous norms and discrete norms. This enables us to study the behavior of the eigenvalues of $\mathbf{E}_{T,m}$ through studying the behavior of the eigenvalues of the variational eigenvalue problem, which we describe next.

Let $e_1 \leq \cdots \leq e_n$ be the eigenvalues of $\mathbf{E}_{T,m}$ in ascending order. Clearly e_i are non-negative real numbers since the matrix $\mathbf{E}_{T,m}$ is semi-positive definite, i.e., $\mathbf{f}^T\mathbf{E}_{T,m}\mathbf{f} = n|f|^2_{T,m} \geq 0$. Next we study the behavior of these eigenvalues and show that they can be bounded by the discrete spectrum of the differential operator $(-\Delta)^m$, where Δ is the Laplacian on $\mathbb{R}^d$.

Theorem 4.3.1. *Suppose Ω satisfies Assumption 1 and T satisfies Assumption 3. Then there exist constants $C_3, C_4 > 0$ such that*

$$C_3\rho_j \leq e_j \leq C_4\rho_j,$$

where $\rho_1 \leq \rho_2 \leq \cdots \leq \rho_n$ are the first n eigenvalues of the variational eigenvalue problem

$$(\phi, \psi)_{\Omega,m} = \rho(\phi, \psi)_{\Omega,0}, \quad \forall\, \psi \in D^{-m}L_2(\Omega).$$

Based on Theorem 4.3.1, we have the following useful result.

Theorem 4.3.2. *Suppose Ω satisfies Assumption 1. Let $\{e_1 \leq \cdots \leq e_n\}$ be the eigenvalues of $\mathbf{E}_{T,m}$ in ascending order. Then there exist constants $C_5, C_6 > 0$ such that for $m(d) = \frac{(d+m-1)!}{d!(m-1)!} < j \leq n$ we have*

$$C_5 j^{\frac{2m}{d}} \leq e_j \leq C_6 j^{\frac{2m}{d}}. \tag{4.3.9}$$

Next we study the behavior of the eigenvalues of the $\mathbf{M}$ matrix in (4.2.13) based on the results in Theorem 4.3.2. Before that, we present the error associated with the estimated derivatives.

Theorem 4.3.3. *Suppose T satisfies Assumptions 3 and 5 and f satisfies Assumption 4. Then there exist constants $\bar{C}_i$ such that*

$$\|\hat{\mathbf{h}}_i - D^\alpha f(X_i)\| \leq \bar{C}_i |f|_{(m,1)}\delta_{\max},\ i = 1, \cdots, n,$$

where $\mathbf{h}_i$ is given in (4.2.11).

We are now ready to present the results on the eigenvalues of the $\mathbf{M}$ matrix in the CDS estimator.

Theorem 4.3.4. *Assume the same conditions as in Theorems 4.3.2 and 4.3.3. Let $\mu_1 \leq \cdots \leq \mu_n$ be the eigenvalues of the matrix $\mathbf{M}$ in (4.2.13). Then there exist constants $C_7, C_8 > 0$ such that for $m(d) < j \leq n$ we have*

$$C_7 j^{\frac{2m}{d}} \leq \mu_j \leq C_8 j^{\frac{2m}{d}}.$$

Based on Theorem 4.3.4, we show in the following theorem that, with probability one, the CDS estimator converges to the true function with an optimal rate.

Theorem 4.3.5. *Suppose Assumptions 1, 3, 4, and 5 are fulfilled. Let $\hat{\mathbf{f}}_n(\lambda) = \mathbf{A}_n(\lambda)\mathbf{y} = (\mathbf{I}_n + \lambda\mathbf{M})^{-1}\mathbf{y}$ be the CDS estimator. Denote $r_n(\lambda) = n^{-1}\|\hat{\mathbf{f}}_n(\lambda) - \mathbf{f}\|^2$. As $n \to \infty$ and $\lambda \sim n^{-2m/(2m+d)}$ is chosen, we have with probability one*

$$E[r_n(\lambda)] = O(n^{-\frac{2m}{2m+d}}).$$

4.3.3 Asymptotic optimality of GCV

In this subsection we will show that our proposed estimator satisfies some general conditions (Section 4.3.3.1) and then prove the asymptotic optimality of GCV under these conditions (Section 4.3.3.2).

4.3.3.1 General conditions

Let $\hat{\mathbf{f}}_n(\lambda) = \mathbf{A}_n(\lambda)\mathbf{y} = (\mathbf{I}_n + \lambda\mathbf{M})^{-1}\mathbf{y}$ be the CDS estimator with order m and denote $r_n(\lambda) = n^{-1}\|\hat{\mathbf{f}}_n(\lambda) - \mathbf{f}\|^2$. The asymptotic optimality of GCV is defined as

$$\frac{r_n(\hat{\lambda}_G)}{\inf\limits_{\lambda \in \mathbb{R}_+} r_n(\lambda)} \longrightarrow_p 1, \tag{4.3.10}$$

where $\longrightarrow_p$ means convergence in probability. As a first step, we show our estimator fulfills the following three conditions.

(A.1) $\inf\limits_{\lambda \in \mathbb{R}_+} nE[r_n(\lambda)] \to \infty$.

(A.2) There exists a sequence $\{\lambda_n\}$ such that $r_n(\lambda_n) \longrightarrow_p 0$.

(A.3) Let $0 \leq \kappa_1 \leq \cdots \leq \kappa_n$ be the eigenvalues of $\mathbf{K}_n(\lambda) = \lambda\mathbf{M}$. For any ℓ such that $\dfrac{\ell}{n} \to 0$, then $\dfrac{\left(n^{-1}\sum_{i=\ell+1}^{n}\kappa_i^{-1}\right)^2}{n^{-1}\sum_{i=\ell+1}^{n}\kappa_i^{-2}} \to 0$ as $n \to \infty$.

The condition (A.1) states that the convergence rate of the risk function to zero should be lower than $O(n^{-1})$. Otherwise the estimates may possess unattainably small risk. The convergence rate $O(n^{-1})$ usually applies to linear regression, while here the scenario is purely nonparametric. Let $\mathscr{P}_{m-1}^d$ be the set of any polynomial on $\mathbb{R}^d$ of degree $\leq m-1$. Actually from the behavior of eigenvalues as shown in Theorem 4.3.4, it is not difficult to verify that our proposed model meets the condition (A.1) except for $f \in \mathscr{P}_{m-1}^d$.

Lemma 4.3.4. *If $f \notin \mathscr{P}_{m-1}^d$, the estimator $\hat{\mathbf{f}}_n(\lambda)$ from our CDS model satisfies*

$$\inf_{\lambda\in\mathbb{R}_+} nE[r_n(\lambda)] \to \infty.$$

Proof of the above lemma is in appendix.

Lemma 4.3.5. *Under condition (A.1), we have in probability*

$$\sup_{\lambda>0}\left|\frac{r_n(\lambda)}{E[r_n(\lambda)]} - 1\right| \to 0. \tag{4.3.11}$$

Proof of the above lemma is in appendix.

The condition (A.2) shows that the risk function $r_n(\lambda_n)$ converge to zero in probability with appropriate sequence $\{\lambda_n\}$. Obviously, the conclusion of condition (A.2) can be easily derived from Theorem 4.3.5 and Lemma 4.3.5.

The condition (A.3) gives a ratio

$$\frac{\left(n^{-1}\sum_{i=\ell+1}^{n}\kappa_i^{-1}\right)^2}{n^{-1}\sum_{i=\ell+1}^{n}\kappa_i^{-2}}, \tag{4.3.12}$$

which is defined on the eigenvalues of $\mathbf{K}_n(\lambda) = \lambda\mathbf{M}$ and often plays an important role in the asymptotic analysis. This condition is essentially saying that the largest eigenvalues of $\mathbf{M}$ should converge to infinity with rate faster than $O(n)$.

Lemma 4.3.6. *In our model, for any ℓ such that $\frac{\ell}{n} \to 0$ and $\kappa_{\ell+1} > 0$, the ratio of* (4.3.12) *converges to zero as n goes to infinity.*

Proof of the above lemma is in appendix.

4.3.3.2 Asymptotic optimality theorems

Under the aforementioned three conditions, we are now ready to prove the asymptotic optimality of GCV. The results are summarized in the following lemmas and the final main theorem.

Lemma 4.3.7. *Under the condition (A.2), we have*

$$n^{-1}\mathrm{tr}[\mathbf{I}_n - \mathbf{A}_n(\lambda_n)] \to 1, \tag{4.3.13}$$

and

$$n^{-1}\|(\mathbf{I}_n - \mathbf{A}_n(\lambda_n))\mathbf{y}\|^2 \to \sigma^2. \tag{4.3.14}$$

Lemma 4.3.8. *Under the condition (A.3), for λ_n such that $r_n(\lambda_n) \to 0$, we have*

$$\frac{\left(n^{-1}\mathrm{tr}[\mathbf{A}_n(\lambda_n)]\right)^2}{n^{-1}\mathrm{tr}[\mathbf{A}_n(\lambda_n)^2]} \to 0. \tag{4.3.15}$$

Lemma 4.3.9. *For any $\hat{\lambda}$ such that $r_n(\hat{\lambda}) \to 0$ and*

$$\frac{\left(n^{-1}\mathrm{tr}[\mathbf{A}_n(\hat{\lambda})]\right)^2}{n^{-1}\mathrm{tr}[\mathbf{A}_n(\hat{\lambda})^2]} \to 0, \tag{4.3.16}$$

under the condition (A.1) we have

$$\frac{\left|\mathrm{SURE}_n(\hat{\lambda}) - \tilde{r}_n(\hat{\lambda}) - n^{-1}\|\varepsilon\|^2 + \sigma^2\right|}{r_n(\hat{\lambda})} \longrightarrow_p 0, \tag{4.3.17}$$

and

$$\frac{n^{-1}\|\tilde{\mathbf{f}}_n(\hat{\lambda}) - \hat{\mathbf{f}}_n(\hat{\lambda})\|^2}{r_n(\hat{\lambda})} \longrightarrow_p 0, \tag{4.3.18}$$

where $\mathrm{SURE}_n(\lambda) = \sigma^2 - \sigma^4 \frac{\left(n^{-1}\mathrm{tr}[\mathbf{I}_n - \mathbf{A}_n(\lambda)]\right)^2}{n^{-1}\|(\mathbf{I}_n - \mathbf{A}_n(\lambda))\mathbf{y}\|^2}$, $\tilde{\mathbf{f}}_n(\lambda) = \mathbf{y} - \sigma^2 \frac{\mathrm{tr}[\mathbf{I}_n - \mathbf{A}_n(\lambda)]}{\|(\mathbf{I}_n - \mathbf{A}_n(\lambda))\mathbf{y}\|^2}(\mathbf{I}_n - \mathbf{A}_n(\lambda))\mathbf{y}$ *and* $\tilde{r}_n(\lambda) = n^{-1}\|\tilde{\mathbf{f}}_n(\lambda) - \mathbf{f}\|^2$.

Lemma 4.3.10. *Under conditions (A.2) and (A.3),* $\hat{\mathbf{f}}_n(\hat{\lambda}_G)$ *is consistent, i.e.,* $r_n(\hat{\lambda}_G) \to 0$*, where* $\hat{\lambda}_G$ *is chosen by GCV.*

Proof of the Lemma 4.3.10 is left in appendix.

Theorem 4.3.6. *Under conditions (A.1), (A.2) and (A.3),* $\hat{\mathbf{f}}_n(\hat{\lambda}_G)$ *is asymptotically optimal, where* $\hat{\lambda}_G$ *is the GCV choice.*

Proof is in appendix.

4.4 Numerical experiments

In this section, we present the numerical results for CDS for $m = 2$. Specifically, we compare CDS against soap film ([72]) and TPS ([17], [64]) over both regular and irregular domains and demonstrate the advantage of CDS. A complexity analysis is carried out in Section 4.4.1. Comparisons with other functional estimation methods (i.e., soap film and the thin plate spline) are conducted over regular and irregular domains in Section 4.4.2.

4.4.1 Complexity analysis

An important feature of the CDS method is the sparsity of the matrix $\mathbf{M}$. Our implemented algorithm benefits greatly from the sparsity and it is very efficient both in spatial and temporal cost. For a data set of size n, the number of nonzeros of $\mathbf{M}$ is strictly less than $(k+1)^2 n$ in theory and empirically $3kn$ as shown in Fig. 4.4.1. Furthermore, $\mathbf{M}$ can be permutated to a band matrix by the symmetric reverse Cuthill-Mckee ordering [12, 33] with a complexity of $O(nk \log k)$ [5, 21]. The symmetric reverse Cuthill-McKee ordering of $\mathbf{M}$ returns a permutation $\mathbf{r} = \text{symrcm}(\mathbf{M})$ such that $\mathbf{M}(\mathbf{r}, \mathbf{r})$ tends to have its nonzero elements closer to the diagonal (see, e.g., Fig. 4.4.2). This is a good pre-ordering for LU or Cholesky factorization of matrices that come from long, skinny problems, such as our case. Moreover, the eigenvalues of $\mathbf{M}(\mathbf{r}, \mathbf{r})$ are the same as those of $\mathbf{M}$. Suppose p is the bandwidth of the obtained matrix $\mathbf{M}(\mathbf{r}, \mathbf{r})$. According to [23], if a band LDL^T decomposition procedure is coupled with an appropriate band triangular solver routine, then approximately $p^2 n + 8pn + n$ flops

and no square roots are required to solve the linear system $(\mathbf{I}_n + \lambda\mathbf{M}(\mathbf{r},\mathbf{r}))\mathbf{f}(\mathbf{r}) = \mathbf{y}(\mathbf{r})$ with given parameter λ. The total number of floating-point operations is proportional to pn^2 for the symmetric banded eigen-decomposition problem eig($\mathbf{M}(\mathbf{r},\mathbf{r})$), which computes all eigenvalues of a real symmetric band matrix. From Fig. 4.4.3, we empirically observe that the bandwidth p is $O((kn)^{0.5})$. Therefore the complexity of CDS is at most $O(n^{2.5})$.

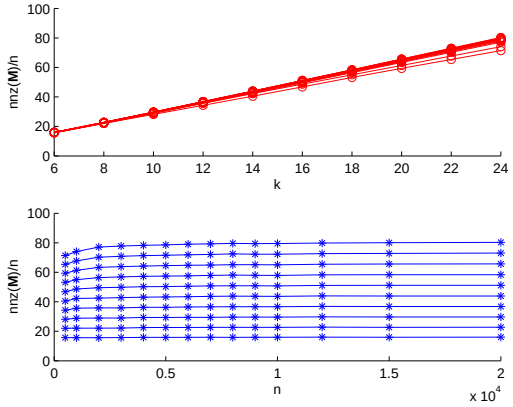

Figure 4.4.1: The number of non-zero elements of matrix $\mathbf{M}$ as functions of k (upper panel) and n (lower panel). It shows that the number of non-zeros is approximately $3kn$.

4.4.2 Numerical comparison with other state-of-the-art methods

In this section, we compare CDS against soap film [72] and thin plate spline [17, 65] using several examples over both irregular and regular domains. Note that soap film was proposed to deal with functional estimation on irregular domains and shown to have better performance than other methods, such as finite element spline [52] and thin plate spline. In this paper, we choose not to compare with finite element spline directly. Readers can infer the comparison outcome by combining the comparison in this paper, and those in [72]. Note that soap film requires a tedious step to handle the boundary in its smoother construction, while CDS does not. Thin plate spline generally outperforms soap film on regular domains, e.g., squares. In our experiments, CDS was implemented in Matlab R2010a, while soap film and thin plate spline were implemented in R2.11.1 [71, 70]. The smoothing parameters were selected by GCV. Typically we set $k = 8$, which is the number of nearest neighbors

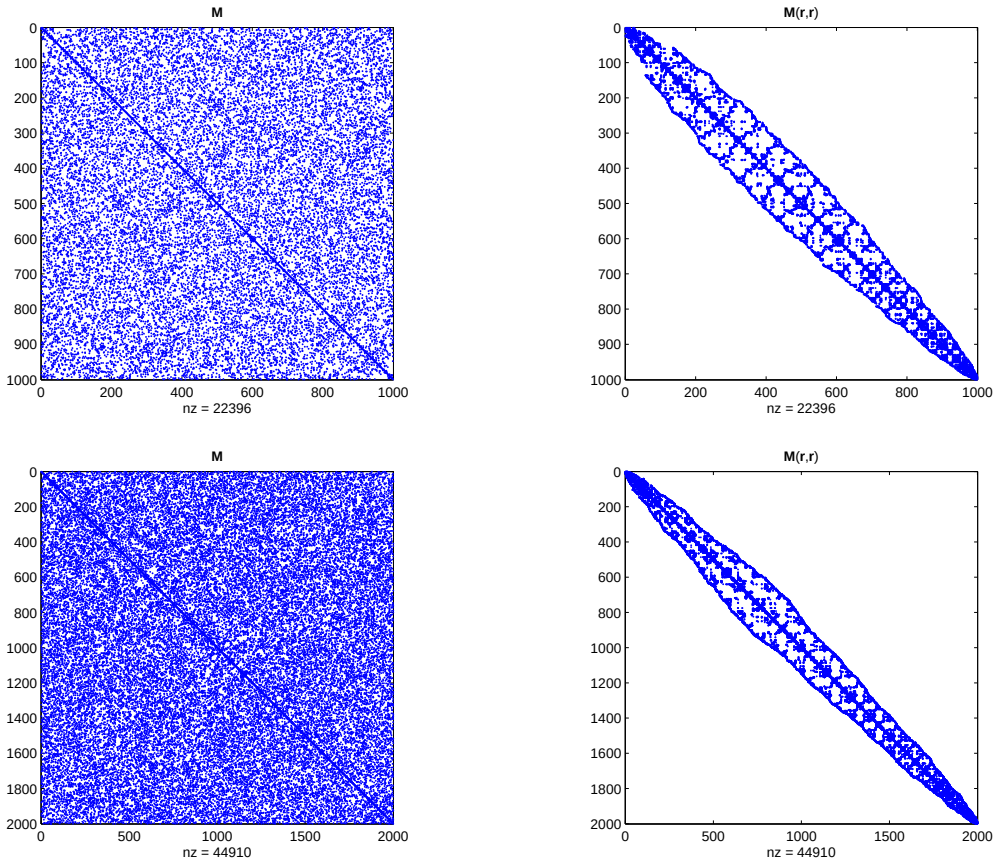

Figure 4.4.2: The symmetric sparse matrix $\mathbf{M}$ (left column) and its re-ordered counterpart $\mathbf{M}(\mathbf{r}, \mathbf{r})$ (right column), after the reverse Cuthill-Mckee ordering. First row: $n = 1000$. Second row: $n = 2000$.

specified for the local estimate of the penalty term (in this case the Hessian) in CDS. In the simulation, we use 32 knots and a basis dimension of 40 for soap film, and a basis dimension of 500 for thin plate spline (actually the thin plate regression spline by [69]).

For the comparison on irregular domain, we use the modified horseshoe example presented in [72], as shown in Fig. 4.4.4. The test function ranges from -4 to 4. We generate true function values with three different sample sizes: 1000, 2000, and 5000. The generated function values were contaminated by Gaussian noise with standard deviation of 0.1, 1, and 10, respectively. For each combination of sample size and noise level, we repeat the experiment for 30 times and calculate the root-mean-square error (RMSE) of CDS, soap film, and thin plate spline, respectively, at the sampled points. Fig. 4.4.5 presents the simulation

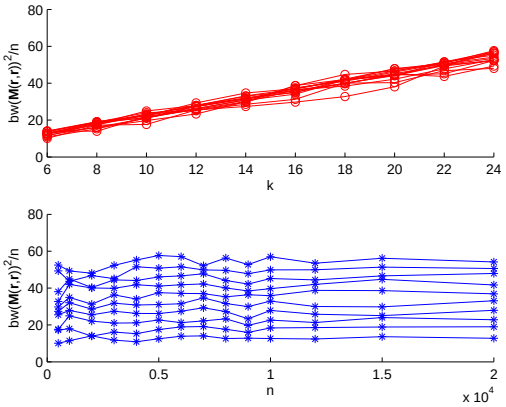

Figure 4.4.3: The bandwidth of matrix $\mathbf{M}(\mathbf{r}, \mathbf{r})$ after the reverse Cuthill-Mckee ordering, as functions of k (upper panel) and n (lower one). The value on the vertical axis is the squared bandwidth divided by n. The upper panel indicates that the aforementioned quantity is a linear function of k, while the lower panel implies that the same quantity is a constant for n. Overall, this empirical evidence hints that bandwidth is approximately $O(\sqrt{kn})$.

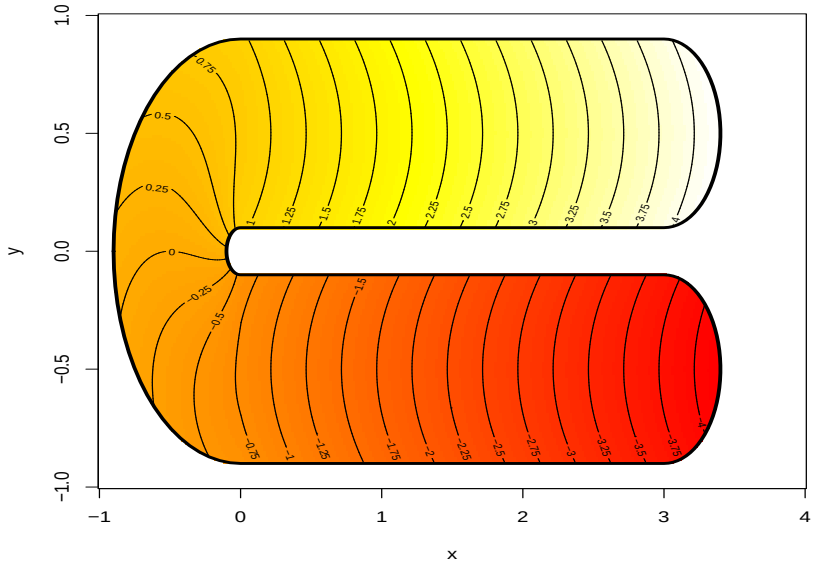

Figure 4.4.4: The horseshoe domain and test function used in [72].

results, which demonstrates that CDS and soap film have comparable performance, while CDS is getting better performance faster as the sample size increases.Thin plate spline's poor performance indicates its inability to deal with irregular domains, like the horseshoe example here.

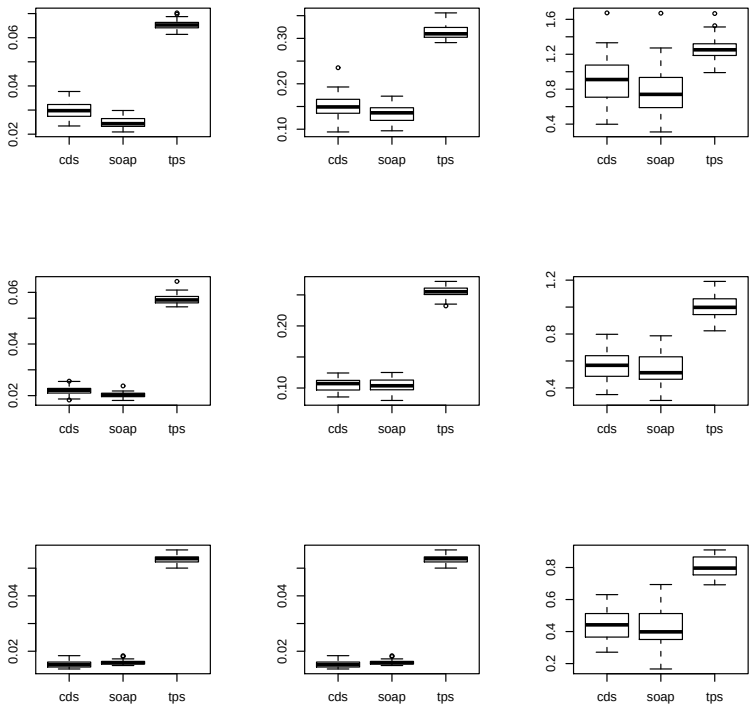

Figure 4.4.5: The RMSE performance of the CDS, Soap film, and TPS on the horseshoe domain. The first, second, and third rows are for $n = 1000, 2000, 5000$, respectively. The left to right columns are associated with noise levels of 0.1,1 and 10.

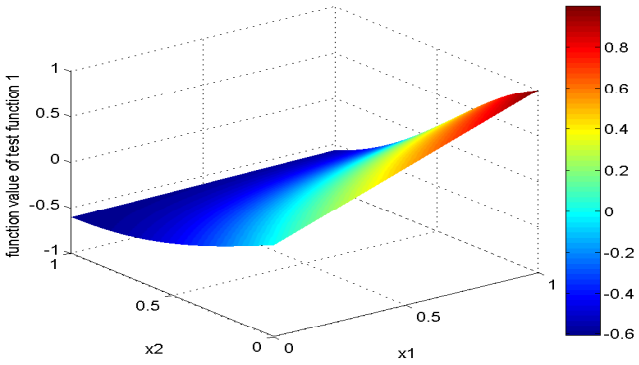

Figure 4.4.6: Contour plot of a test function.

We also use two examples on regular domains to compare the three methods. The

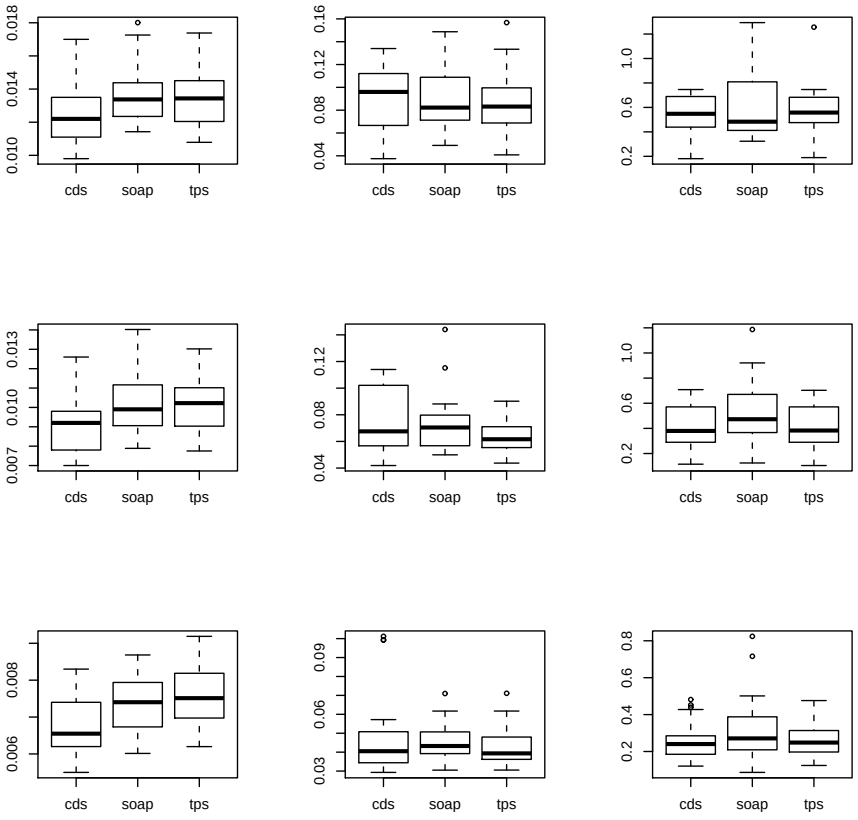

Figure 4.4.7: The RMSE performance of CDS, Soap film, and TPS on a regular domain. The first, second, and third rows are for $n = 1000, 2000$, and 5000, respectively. Left to right columns correspond to noise levels of 0.1,1 and 10.

function in the first example (Fig 4.4.6) is defined as

$$f(x_1, x_2) = -x_2 \exp(-0.5x_2^2) + \frac{x_1}{1 + x_2^2} \cos(1.5x_2).$$

As shown in Fig. 4.4.7, three methods have similar performance. The function in this example has a small total variation and is relatively easy to estimate. We use a function that has many dramatic fluctuations in the second example. One of its level-set resembles the letter 'R' (Fig. 4.4.8). From the results that are summarized in Fig. 4.4.9, one can see that CDS and thin plate spline have significantly better performance than soap film. This may imply that soap film does not work well for very wiggly functions.

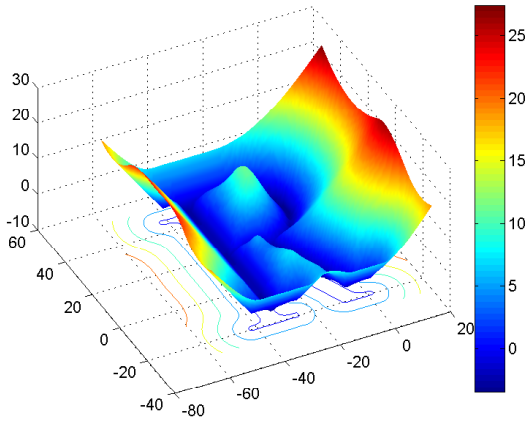

Figure 4.4.8: The second test function for a regular domain. A level-set of this function resembles the letter 'R'.

In summary, CDS is a practical method for functional estimation on both regular domains and irregular domains. The advantage of CDS becomes more pronounced as the sample size increases.

4.5 Concluding remarks

We have proposed a new nonparametric functional estimation method called CDS. The new estimator is motivated by the theoretical difficulty of finding a set of basis functions as well as the computational burden in finding the desired estimator when the domain of inputs is irregular within the smoothing splines framework. The essential idea is to replace the penalty term with a novel estimator based on local least squares.

We have shown that, with some regularity assumptions, CDS enjoys the same convergence rate as the D^m-smoothing splines. However, CDS is easier to understand and faster to compute. In the numerical experiments, we presented a few examples with $m = 2$. From the numerical results, we can see that CDS performs well for both regular and irregular domains. Specifically, CDS has comparable performance with soap and is much better than TPS for the horseshoe example. Note that the horseshoe example has an irregular domain. For the R example, CDS is comparable to TPS and much better than soap. This shows that

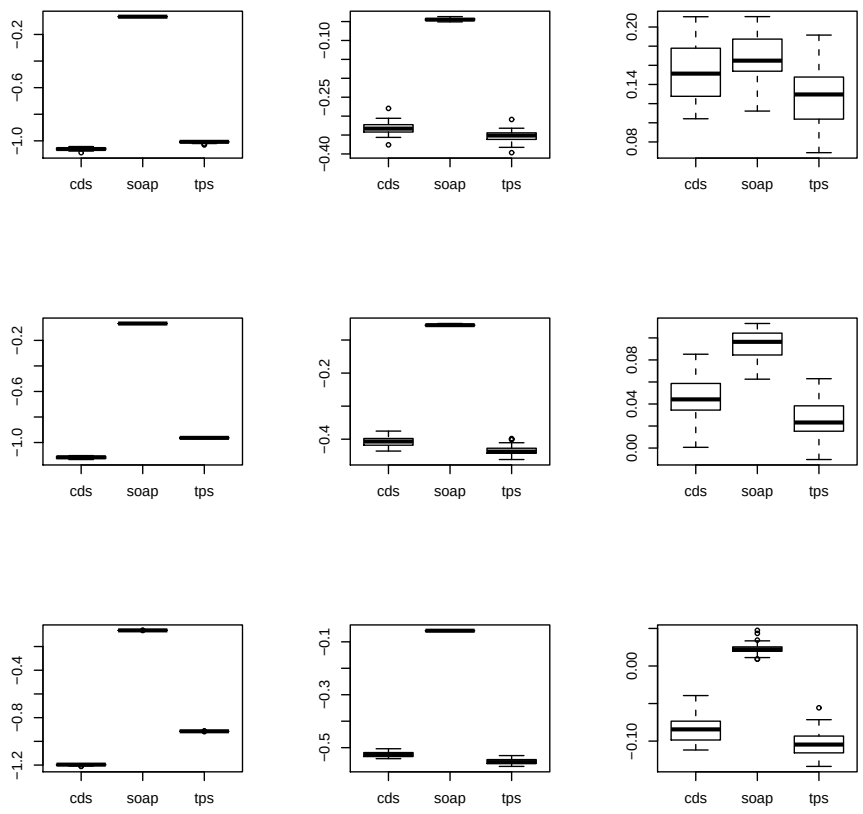

Figure 4.4.9: The RMSE performance of the CDS, Soap film, and TPS on the test function, who has a shape 'R' level set. For visualization, the RMSE is scaled by $\log_{10}$. The first, second, and third rows are for $n = 1000, 2000$, and 5000, respectively. The left to right columns correspond to the noise levels of 0.1,1 and 5.

soap does not perform well when the underlying function is very wiggle.

For the future research directions, we can extend our method to other penalty terms other than those shown in the paper. We need also consider faster computation of CDS and analyze the effect of the choice of number of nearest neighbors in the estimation. The reason that CDS works well for irregular domains is that it is essentially a local smoothing method. This is reflected in the sparsity of the matrix $\mathbf{M}$. CDS only uses local information to estimate the function value at a specific location. This intuition can help to derive the theoretical properties of CDS for those irregular domains that do not satisfy the regularity

conditions in the chapter. Here is the hunch. Suppose the irregular domain can be partitioned into finite number of pieces, each of which is like a regular domain. Then CDS is essentially estimating the underlying function on each piece separately. Since we have established the theoretical properties of CDS for regular domains, we should be able to combine the results across different pieces and derive the overall performance of CDS for the entire irregular domain.

4.6 Appendix

4.6.1 Proof of Lemma 4.3.1

Proof. Suppose that V_d is the volume of unit sphere in d-dimensional space. We have

$$nV_d\delta_{\min}^d \leq \mathrm{Vol}(\Omega),$$

and thus

$$\delta_{\max}^d \leq n^{-1}\frac{\mathrm{Vol}(\Omega)}{V_d}\frac{\delta_{\max}^d}{\delta_{\min}^d} = n^{-1}\frac{\mathrm{Vol}(\Omega)}{V_d}B_0^d = O(n^{-1}). \tag{4.6.1}$$

Therefore $n\delta_{\max}^d$ is bounded from above.

Let u be the function such that

$$\frac{1}{n}\mathbf{u}^T\mathbf{E}_{T,m}\mathbf{u} = |u|_{T,m}^2 = \min_{\substack{\phi \in D^{-m}L_2(\Omega) \\ \phi(X_i)=u_i,\, i=1,\cdots,n}} |\phi|_{T,m}^2 \tag{4.6.2}$$

where $\mathbf{u} = (u_1, \cdots, u_n)^T$ is the eigenvector of $\mathbf{E}_{T,m}$ corresponding to the largest eigenvalue, i.e., $\mathbf{E}_{T,m}\mathbf{u} = e_n\mathbf{u}$. We define a compactly supported radial basis function

$$w(s) = \begin{cases} e^{-\|\mathbf{s}\|/(1-\|\mathbf{s}\|)}, & 0 \leq \|\mathbf{s}\| < 1 \\ 0, & \|\mathbf{s}\| \geq 1 \end{cases}$$

and specify an interpolant

$$\phi(X) = \sum_{i=1}^{n} u_i w_i(X)$$

where $w_i(X) = w(\frac{X-X_i}{\delta_{\min}})$. By the definition of $\delta_{\min}$, it is easy to see that $\phi(X_i) = u_i, i = 1, \cdots, n$. Moreover, we have for $\alpha \in \mathbb{Z}_+^d$

$$D^\alpha w_i(X_j) = 0, \quad \forall i \neq j$$

and with $|\alpha| = m$

$$D^\alpha w_j(X_j) = \delta_{\min}^{-m} D^\alpha w(\mathbf{0}).$$

Hence, we have

$$\begin{aligned}
|u|_{T,m}^2 &\le |\phi|_{T,m}^2 \\
&= \frac{1}{n}\sum_{j=1}^{n}\Big(\sum_{|\alpha|=m}\tfrac{m!}{\alpha!}|D^\alpha\phi(X_j)|^2\Big) \\
&= \frac{1}{n}\sum_{j=1}^{n}\Big(\sum_{|\alpha|=m}\tfrac{m!}{\alpha!}|\sum_{i=1}^{n}u_i D^\alpha w_i(X_j)|^2\Big) \\
&= \frac{1}{n}\sum_{j=1}^{n}\Big(\sum_{|\alpha|=m}\tfrac{m!}{\alpha!}u_j^2|D^\alpha w_j(X_j)|^2\Big) \\
&= \frac{1}{n}\sum_{j=1}^{n}u_j^2\Big(\sum_{|\alpha|=m}\tfrac{m!}{\alpha!}|D^\alpha w(\mathbf{0})|^2\Big)\delta_{\min}^{-2m},
\end{aligned}$$

which implies that

$$e_n \le C(w,m)\delta_{\min}^{-2m}$$

by denoting the constant $C(w,m) = \sum_{|\alpha|=m}\frac{m!}{\alpha!}|D^\alpha w(\mathbf{0})|^2$. Finally we get

$$\delta_{\max}^{2m} e_n \le C(w,m)\frac{\delta_{\max}^{2m}}{\delta_{\min}^{2m}} = C(w,m)B_0^{2m}$$

and prove that $\delta_{\max}^{2m} e_n$ is bounded from above. □

4.6.2 Proof of Lemma 4.3.2

Proof. According to Theorem 3.3 in [63], there exist constant $C(d,m,\Omega,B_0) > 0$ and $\delta_0 > 0$ such that for $\delta_{\max} \le \delta_0$,

$$|f|_{T,0}^2 \le C(d,m,\Omega,B_0)(|f|_{\Omega,0}^2 + \delta_{\max}^{2m}|f|_{\Omega,m}^2).$$

Since $|f|_{T,m}^2 \ge \underline{B}|f|_{\Omega,m}^2$, we have

$$\frac{|f|_{T,m}^2}{|f|_{T,0}^2} \ge \frac{\underline{B}|f|_{\Omega,m}^2}{C(d,m,\Omega,B_0)(|f|_{\Omega,0}^2 + \delta_{\max}^{2m}|f|_{\Omega,m}^2)} = \frac{|f|_{\Omega,m}^2}{C_1(|f|_{\Omega,0}^2 + \delta_{\max}^{2m}|f|_{\Omega,m}^2)},$$

where $C_1 = C(d,m,\Omega,B_0)/\underline{B}$. □

4.6.3 Proof of Lemma 4.3.3

Proof. According to Theorem 3.4 in [63], there exist constant $C(d, m, \Omega, B_0) > 0$ and $\delta_0 > 0$ such that for $\delta_{\max} \leq \delta_0$,

$$|f|^2_{\Omega,0} \leq C'(d, m, \Omega, B_0)(|f|^2_{T,0} + \delta^{2m}_{\max}|f|^2_{\Omega,m}).$$

Since $\underline{B}|f|^2_{\Omega,m} \leq |f|^2_{T,m} \leq \overline{B}|f|^2_{\Omega,m}$, we have

$$\frac{|f|^2_{\Omega,m}}{|f|^2_{\Omega,0}} \geq \frac{|f|^2_{T,m}/\overline{B}}{C'(d, m, \Omega, B_0)(|f|^2_{\Omega,0} + \delta^{2m}_{\max}|f|^2_{T,m}/\underline{B})} \geq \frac{|f|^2_{T,m}}{C_2(|f|^2_{T,0} + \delta^{2m}_{\max}|f|^2_{T,m})},$$

where $C_2 = \overline{B}C'(d, m, \Omega, B_0)\max(1, 1/\underline{B})$. □

4.6.4 Proof of Theorem 4.3.1

Proof. From Lemma 4.3.2 we get

$$\frac{|\phi|^2_{T,m}}{|\phi|^2_{T,0}} \geq \frac{|\phi|^2_{\Omega,m}}{C_1(|\phi|^2_{\Omega,0} + \delta^{2m}_{\max}|\phi|^2_{\Omega,m})}$$

for any $\phi \in D^{-m}L_2(\Omega)$ with $|\phi|^2_{T,0} \neq 0$. Thus

$$e_j \geq \frac{1}{C_1}\vartheta_j,$$

where $\vartheta_1 \leq \cdots \leq \vartheta_n$ are the first n eigenvalues of the variational eigenvalue problem

$$|\phi|^2_{\Omega,m} = \vartheta \cdot (|\phi|^2_{\Omega,0} + \delta^{2m}_{\max}|\phi|^2_{\Omega,m}),$$

which implies

$$\vartheta_j = \frac{\rho_j}{1 + \delta^{2m}_{\max}\rho_j}, j = 1, \cdots, n.$$

Note that $\delta^{2m}_{\max}\rho_j$ is bounded from above, since $\rho_j \sim j^{\frac{2m}{d}}$ according to Theorem 14.6 in [1] and $\delta^{2m}_{\max} = O(n^{-2m/d})$ from (4.6.1). So there exists $C_3 > 0$ such that $\frac{1}{C_1(1+\delta^{2m}_{\max}\rho_j)} \geq C_3$, and we have

$$e_j \geq C_3\rho_j.$$

On the other hand, using Lemma 4.3.3

$$\frac{|\phi|^2_{\Omega,m}}{|\phi|^2_{\Omega,0}} \geq \frac{|\phi|^2_{T,m}}{C_1(|\phi|^2_{T,0} + \delta^{2m}_{\max}|\phi|^2_{T,m})}$$

we have

$$\rho_j \geq \frac{1}{C_2}\nu_j,$$

where $\nu_1 \leq \cdots \leq \nu_n$ are the first n eigenvalues of the variational eigenvalue problem

$$|\phi|^2_{T,m} = \nu \cdot (|\phi|^2_{T,0} + \delta^{2m}_{\max}|\phi|^2_{T,m}),$$

which implies

$$\nu_j = \frac{e_j}{1 + \delta^{2m}_{\max} e_j}, j = 1, \cdots, n.$$

So there exists $C_4 > 0$ such that

$$e_j \leq C_2(1 + \delta^{2m}_{\max} e_j)\rho_j \leq C_2(1 + \delta^{2m}_{\max} e_n)\rho_j \leq C_4\rho_j,$$

since $\delta^{2m}_{\max} e_n$ is bounded according to Lemma 4.3.1. □

4.6.5 Proof of Lemma 4.3.2

Proof. According to Theorem 4.3.1, it suffices to prove that the eigenvalues $\rho_1 \leq \rho_2 \leq \cdots$ satisfy the type of relationship in (4.3.9).

By using integration by parts, we observe that $\rho_1 \leq \rho_2 \leq \cdots$ are the eigenvalues of the differential operator $(-\Delta)^m$ which has discrete spectrum contained in the nonnegative real axis. We can then apply Theorem 14.6 in [1] to get

$$\rho_j \sim j^{\frac{2m}{d}}, \quad j > m(d).$$

This concludes the proof. □

4.6.6 Proof of Theorem 4.3.3

Proof. Recall the Taylor expansion of f can be written as follows

$$f(X) = f(X_i) + \sum_{|\alpha|=1}^{m} D^\alpha f(X_i) B_\alpha(X, X_i) + R_{m+1}(X, X_i), \tag{4.6.3}$$

where $R_{m+1}(X, X_i)$ is the remainder term. From [67], for any function in $C^{m,1}(\Omega)$, $R_{m+1}(X, X_i)$ has the following property: there exists a constant $\bar{C}_1$ such that for every $X \in \Omega$ we have

$$|R_{m+1}(X, X_i)| \leq \bar{C}_1 |f|_{(m,1)} \|X - X_i\|^{m+1}, \tag{4.6.4}$$

where the semi-norm $|f|_{(m,1)}$ is defined as

$$|f|_{(m,1)} = \sup \left\{ \frac{|D^\alpha f(X) - D^\alpha f(Y)|}{\|X - Y\|} : X \neq Y \in \Omega, |\alpha| = m \right\}.$$

Since we use the local least squares method to estimate $D^m f$, according to Theorem 7 in [76], there exists constant $\bar{C}_2$ such that

$$\|\hat{\mathbf{h}}_i - D^\alpha f(X_i)\| \leq \bar{C}_2 h^{-|\alpha|} \|R_{m+1}(X_i)\|, \mathrm{i} = 1, \cdots, n$$

where $h = \max_{X_j \in \mathscr{B}_k(X_i)} \|X_j - X_i\|$ and $R_{m+1}(X_i) = (R_{m+1}(X_j, X_i))_{X_j \in \mathscr{B}_k(X_i)}$. Furthermore, from (4.6.4) we have

$$\|\hat{\mathbf{h}}_i - D^m f(X_i)\| \leq \bar{C}_2 h^{-m} \bar{C}_1 |f|_{(m,1)} h^{m+1} \leq \bar{C} |f|_{(m,1)} \delta_{\max},$$

where $\bar{C} = 2k\bar{C}_1\bar{C}_2$ and the last inequality is based on the fact $h \leq 2k\delta_{\max}$. □

4.6.7 Proof of Theorem 4.3.4

Proof. Without loss of generality, let $\mathbf{f}$ be the vector of function values such that $|f|^2_{T,0} = \frac{1}{n}\mathbf{f}^T\mathbf{f} = 1$. Based on Theorem 4.3.3, we have

$$\begin{aligned}
\left| \frac{\frac{1}{n}\mathbf{f}^T\mathbf{M}\mathbf{f}}{\frac{1}{n}\mathbf{f}^T\mathbf{f}} - \frac{|f|^2_{T,m}}{|f|^2_{T,0}} \right| &= \frac{\left| \frac{1}{n}\mathbf{f}^T\mathbf{M}\mathbf{f} - |f|^2_{T,m} \right|}{|f|^2_{T,0}} \\
&\leq \frac{1}{n} \sum_{i=1}^{n} |\|\hat{\mathbf{h}}_i\|^2 - \|D^m f(X_i)\|^2| \\
&\leq \frac{1}{n} \sum_{i=1}^{n} \|\hat{\mathbf{h}}_i - D^m f(X_i)\|^2 \\
&= O(\delta^2_{\max})
\end{aligned}$$

which implies

$$|\mu_j - e_j| = O(n^{-2/d})$$

since the fact $\delta_{\max} = O(n^{-1/d})$ of (4.6.1) in Lemma 4.3.1. Hence, from Theorem 4.3.2, there exist constants $C_7, C_8 > 0$ such that for $m^{(d)} < j \leq n$,

$$C_7 j^{\frac{2m}{d}} \leq \mu_j \leq C_8 j^{\frac{2m}{d}}.$$

□

4.6.8 Proof of Theorem 4.3.5

Proof. By using the bounds of eigenvalues $\mu_j = O(j^{\frac{2m}{d}})$ obtained in Theorem 4.3.4, we have

$$\begin{aligned} E[r_n(\lambda)] &= E[n^{-1}\|\hat{\mathbf{f}}_n(\lambda) - \mathbf{f}\|^2] \\ &= n^{-1}\Big(\mathbf{f}^T (\mathbf{A}_n(\lambda) - \mathbf{I})^2 \mathbf{f} + \sigma^2 \mathrm{tr}[\mathbf{A}_n(\lambda)^2]\Big) \\ &\leq \frac{\lambda}{4n}\mathbf{f}^T\mathbf{M}\mathbf{f} + \frac{\sigma^2}{n}\sum_{j=1}^{n}\frac{1}{(1+\lambda\mu_j)^2} \\ &= O(\lambda) + O(n^{-1}\lambda^{-\frac{d}{2m}}), \end{aligned} \tag{4.6.5}$$

where the last equation is based on the result of Proposition 3.3.1. In particular, if the smoothing parameter is chosen to satisfy $\lambda \sim n^{-2m/(2m+d)}$, then we achieve the convergence rate $E[r_n(\lambda)] = O(n^{-\frac{2m}{2m+d}})$. According to [61], $-\frac{2m}{2m+d}$ is the optimal for nonparametric multivariate functional estimation with the order m in d-dimensional space with some standard regularity assumptions on the domain Ω. Since as $n \to \infty$, Assumption 2 is satisfied with probability one, we know CDS achieves the optimal convergence rate in nonparametric multivariate functional estimation with probability one. □

4.6.9 Proof of Lemma 4.3.4

Proof. Let $0 \leq \mu_1 \leq \cdots \leq \mu_n$ be the eigenvalues of design matrix $\mathbf{M}$, and $\mathbf{u}_j$ the unit eigenvector corresponding to μ_j, $j = 1, \cdots, n$. So we have

$$\begin{aligned} nE[r_n(\lambda)] &= nE[n^{-1}\|\hat{\mathbf{f}}_n(\lambda) - \mathbf{f}\|^2] \\ &= E[(\hat{\mathbf{f}}_n(\lambda) - \mathbf{f})^T(\hat{\mathbf{f}}_n(\lambda) - \mathbf{f})] \\ &= \mathbf{f}^T (\mathbf{A}_n(\lambda) - \mathbf{I})^2 \mathbf{f} + \sigma^2 \mathrm{tr}[\mathbf{A}_n(\lambda)^2] \\ &= \sum_{j=1}^{n}\frac{\lambda^2\mu_j^2}{(1+\lambda\mu_j)^2}b_j^2 + \sigma^2\sum_{j=1}^{n}\frac{1}{(1+\lambda\mu_j)^2}, \end{aligned} \tag{4.6.6}$$

where $b_j = \mathbf{u}_j^T \mathbf{f}$.

If $\lambda \sim O(1)$ or $\lambda \to \infty$ (corresponds to $n \to \infty$), since $\mu_j \sim j^{2m/d}$ there exists j^* such that $\frac{j^*}{n} \to 0$ and $\frac{\lambda\mu_j}{1+\lambda\mu_j} \geq \frac{1}{2}$ for $j > j^*$, then

$$\begin{aligned} nE[r_n(\lambda)] &\geq \sum_{j=1}^{n} \frac{\lambda^2\mu_j^2}{(1+\lambda\mu_j)^2} b_j^2 \\ &\geq \frac{1}{4} \sum_{j>j^*} b_j^2 \\ &\geq \frac{n}{4}|f|_{T,0}^2 - \frac{1}{4} j^* \max\{b_1^2, \cdots, b_{j^*}^2\} \\ &= O(n) \to \infty. \end{aligned}$$

On the other hand, if $\lambda \to 0$ corresponds to $n \to \infty$, we have

$$\begin{aligned} nE[r_n(\lambda)] &\geq \sigma^2 \sum_{j=1}^{n} \frac{1}{(1+\lambda\mu_j)^2} \\ &= O(\lambda^{-\frac{d}{2m}}) \\ &\to \infty, \end{aligned}$$

where the second equation is also based on the Proposition 3.3.1. □

4.6.10 Proof of Lemma 4.3.5

Proof. To get (4.3.11), it suffices to show in probability

$$\sup_{\lambda>0} \frac{n^{-1} \left| \mathbf{f}^T (\mathbf{A}_n(\lambda) - \mathbf{I}_n) \mathbf{A}_n(\lambda) \varepsilon \right|}{E[r_n(\lambda)]} \to 0 \tag{4.6.7}$$

and

$$\sup_{\lambda>0} \frac{n^{-1} \left| \|\mathbf{A}_n(\lambda)\varepsilon\|^2 - \sigma^2 \mathrm{tr}[\mathbf{A}_n(\lambda)^2] \right|}{E[r_n(\lambda)]} \to 0. \tag{4.6.8}$$

According to the Chebyshev inequality, we have for any given $\delta > 0$

$$\begin{aligned} &\Pr\{\frac{n^{-1} \left| \mathbf{f}^T (\mathbf{A}_n(\lambda) - \mathbf{I}_n) \mathbf{A}_n(\lambda) \varepsilon \right|}{E[r_n(\lambda)]} > \delta\} \\ &\leq \delta^{-2} (nE[r_n(\lambda)])^{-2} E\left[(\mathbf{f}^T (\mathbf{A}_n(\lambda) - \mathbf{I}_n) \mathbf{A}_n(\lambda) \varepsilon)^2 \right] \\ &= \delta^{-2} (nE[r_n(\lambda)])^{-2} \sigma^2 \mathrm{tr}\left[\mathbf{A}_n(\lambda)(\mathbf{A}_n(\lambda) - \mathbf{I}_n) \mathbf{f}\, \mathbf{f}^T (\mathbf{A}_n(\lambda) - \mathbf{I}_n) \mathbf{A}_n(\lambda) \right] \\ &= \delta^{-2} (nE[r_n(\lambda)])^{-2} \sigma^2 \|\mathbf{A}_n(\lambda)(\mathbf{A}_n(\lambda) - \mathbf{I}_n)\mathbf{f}\|^2 \\ &\leq \delta^{-2} (nE[r_n(\lambda)])^{-1} \sigma^2 \frac{\|(\mathbf{A}_n(\lambda) - \mathbf{I}_n)\mathbf{f}\|^2}{nE[r_n(\lambda)]} \\ &\leq \delta^{-2} \sigma^2 (nE[r_n(\lambda)])^{-1} \to 0, \end{aligned}$$

since $nE[r_n(\lambda)] \geq \|(\mathbf{A}_n(\lambda) - \mathbf{I}_n)\mathbf{f}\|^2$. Thus (4.6.7) holds in probability.

Again using the Chebyshev inequality, we have for any given $\delta > 0$

$$\begin{aligned}
&\Pr\{\frac{n^{-1}\left|\|\mathbf{A}_n(\lambda)\varepsilon\|^2 - \sigma^2\mathrm{tr}[\mathbf{A}_n(\lambda)^2]\right|}{E[r_n(\lambda)]} > \delta\} \\
&\leq \delta^{-2}(nE[r_n(\lambda)])^{-2}E\left[(\|\mathbf{A}_n(\lambda)\varepsilon\|^2 - \sigma^2\mathrm{tr}[\mathbf{A}_n(\lambda)^2])^2\right] \\
&= \delta^{-2}(nE[r_n(\lambda)])^{-1}\frac{E\left[\|\mathbf{A}_n(\lambda)\varepsilon\|^4\right] - \left(\sigma^2\mathrm{tr}[\mathbf{A}_n(\lambda)^2]\right)^2}{nE[r_n(\lambda)]}.
\end{aligned}$$

Since $nE[r_n(\lambda)] \geq \sigma^2\mathrm{tr}[\mathbf{A}_n(\lambda)^2]$, we only need to show

$$\frac{E\left[\|\mathbf{A}_n(\lambda)\varepsilon\|^4\right] - \left(\sigma^2\mathrm{tr}[\mathbf{A}_n(\lambda)^2]\right)^2}{\sigma^2\mathrm{tr}[\mathbf{A}_n(\lambda)^2]} < \text{Constant}. \tag{4.6.9}$$

Denote $\mathbf{B} = \mathbf{A}_n(\lambda)^2 = (B_{ij})_{n\times n}$, then we have

$$\begin{aligned}
E\left[\|\mathbf{A}_n(\lambda)\varepsilon\|^4\right] &= E\left[(\varepsilon^T\mathbf{B}\varepsilon)^2\right] \\
&= E[(\sum_{i,j} B_{ij}\varepsilon_i\varepsilon_j)(\sum_{i',j'} B_{i'j'}\varepsilon_{i'}\varepsilon_{j'})] \\
&= E[(\sum_{i} B_{ii}\varepsilon_i^2)(\sum_{i'} B_{i'i'}\varepsilon_{i'}^2)] + E[(\sum_{i\neq j} B_{ij}\varepsilon_i\varepsilon_j)(\sum_{i'\neq j'} B_{i'j'}\varepsilon_{i'}\varepsilon_{j'})] \\
&\leq \left(\sum_{i=1}^n B_{ii}\sigma^2\right)^2 + \sum_{i=1}^n B_{ii}^2E[\varepsilon_i^4] + \sum_{i\neq j} B_{ij}^2\sigma^4.
\end{aligned}$$

There exists a constant c such that $E[\varepsilon_i^4] \leq c\sigma^2$ and $\sigma^4 \leq c\sigma^2$, so we get

$$\begin{aligned}
E\left[\|\mathbf{A}_n(\lambda)\varepsilon\|^4\right] &\leq \left(\sum_{i=1}^n B_{ii}\sigma^2\right)^2 + c\sum_{i=1}^n B_{ii}^2\sigma^2 + c\sum_{i\neq j} B_{ij}^2\sigma^2 \\
&= \left(\sum_{i=1}^n B_{ii}\sigma^2\right)^2 + c\sum_{i,j} B_{ij}^2\sigma^2 \\
&= \left(\sigma^2\mathrm{tr}[\mathbf{A}_n(\lambda)^2]\right)^2 + c\sigma^2\mathrm{tr}[\mathbf{A}_n(\lambda)^4] \\
&\leq \left(\sigma^2\mathrm{tr}[\mathbf{A}_n(\lambda)^2]\right)^2 + c\sigma^2\mathrm{tr}[\mathbf{A}_n(\lambda)^2],
\end{aligned}$$

which implies (4.6.9), and immediately leads to (4.6.8) in probability. □

4.6.11 Proof of Lemma 4.3.6

Proof. From the Theorem 4.3.4, i.e., $\mu_i = O(i^{\frac{2m}{d}})$, we get

$$\begin{aligned}
\lim_{n\to\infty} \frac{\left(n^{-1}\sum_{i=\ell+1}^{n}\kappa_i^{-1}\right)^2}{n^{-1}\sum_{i=\ell+1}^{n}\kappa_i^{-2}} &= \lim_{n\to\infty} \frac{\left(\sum_{i=\ell+1}^{n}\mu_i^{-1}\right)^2}{n\sum_{i=\ell+1}^{n}\mu_i^{-2}} \\
&= \lim_{n\to\infty} \frac{\left(\int_{\ell+1}^{n}\mu^{-2m/d}\mathrm{d}\mu\right)^2}{n\int_{\ell+1}^{n}\mu^{-4m/d}\mathrm{d}\mu} \\
&= \lim_{n\to\infty} \frac{(4m-d)d}{(2m-d)^2}\cdot\frac{\left((\ell+1)^{1-\frac{2m}{d}}-n^{1-\frac{2m}{d}}\right)^2}{n\left((\ell+1)^{1-\frac{4m}{d}}-n^{1-\frac{4m}{d}}\right)} \\
&= \lim_{n\to\infty} \frac{(4m-d)d}{(2m-d)^2}\cdot\frac{\ell+1}{n}\cdot\frac{\left(1-(\frac{\ell+1}{n})^{\frac{2m}{d}-1}\right)^2}{\left(1-(\frac{\ell+1}{n})^{\frac{4m}{d}-1}\right)} \\
&= 0.
\end{aligned}$$

□

4.6.12 Proof of Lemma 4.3.7

Proof. From the fact that

$$\sigma^2(n^{-1}\mathrm{tr}[\mathbf{A}_n(\lambda_n)])^2 \le \sigma^2 n^{-1}\mathrm{tr}[\mathbf{A}_n(\lambda_n)^2] \le E[r_n(\lambda_n)] \to 0,$$

we have $n^{-1}\mathrm{tr}[\mathbf{A}_n(\lambda_n)] \to 0$ and then get

$$n^{-1}\mathrm{tr}[\mathbf{I}_n - \mathbf{A}_n(\lambda_n)] \to 1.$$

By the fact $n^{-1}||\varepsilon||^2 \to \sigma^2$ and the Cauchy-Schwartz inequality, we have

$$n^{-1}||(\mathbf{I}_n - \mathbf{A}_n(\lambda_n))\mathbf{y}||^2 = n^{-1}||\varepsilon||^2 + n^{-1}||\mathbf{f} - \hat{\mathbf{f}}_n(\lambda_n)||^2 + \frac{2}{n}|(\mathbf{f} - \hat{\mathbf{f}}_n(\lambda_n))^T\varepsilon| \to \sigma^2.$$

□

4.6.13 Proof of Lemma 4.3.8

Proof. Recall $\mathbf{A}_n(\lambda_n) = (\mathbf{I}_n + \lambda_n\mathbf{M})^{-1} = (\mathbf{I}_n + \mathbf{K}_n(\lambda_n))^{-1}$. We get

$$\frac{\left(n^{-1}\mathrm{tr}[\mathbf{A}_n(\lambda_n)]\right)^2}{n^{-1}\mathrm{tr}[\mathbf{A}_n(\lambda_n)^2]} = \frac{\left(n^{-1}\sum_{i=1}^{n}(1+\kappa_i)^{-1}\right)^2}{n^{-1}\sum_{i=1}^{n}(1+\kappa_i)^{-2}}, \tag{4.6.10}$$

where $0 \leq \kappa_1 \leq \cdots \leq \kappa_n$ are the eigenvalues of $\mathbf{K}_n(\lambda_n)$. Let ℓ be the number holding $\kappa_\ell \leq 1 < \kappa_{\ell+1}$, then we have

$$\sum_{i=1}^{n}(1+\kappa_i)^{-1} \leq \ell + \sum_{i=\ell+1}^{n}\kappa_i^{-1}, \tag{4.6.11}$$

and

$$\sum_{i=1}^{n}(1+\kappa_i)^{-2} \geq \frac{1}{4}\left(\ell + \sum_{i=\ell+1}^{n}\kappa_i^{-2}\right). \tag{4.6.12}$$

To reach (4.3.15), it suffices to show

$$\frac{\left(\frac{\ell}{n} + \frac{1}{n}\sum_{i=\ell+1}^{n}\kappa_i^{-1}\right)^2}{\frac{1}{4}\left(\frac{\ell}{n} + \frac{1}{n}\sum_{i=\ell+1}^{n}\kappa_i^{-2}\right)} \to 0. \tag{4.6.13}$$

On the other hand, $E[r_n(\lambda_n)] \to 0$ since $r_n(\lambda_n)$ is nonnegative, thus we get $n^{-1}\mathrm{tr}[\mathbf{A}_n(\lambda_n)^2] \to 0$ and have $\frac{\ell}{n} \to 0$ due to (4.6.12). So it is not hard to see that (4.6.13) holds under the condition (A.3). □

4.6.14 Proof of Lemma 4.3.9

Proof. We first prove (4.3.17), which can be rewritten as

$$\begin{aligned}
&\frac{2\left|\frac{\sigma^2\mathrm{tr}[\mathbf{I}_n-\mathbf{A}_n(\lambda)]\mathbf{y}^T(\mathbf{I}_n-\mathbf{A}_n(\lambda))\varepsilon}{n\|(\mathbf{I}_n-\mathbf{A}_n(\lambda))\mathbf{y}\|^2} - \frac{\sigma^4(\mathrm{tr}[\mathbf{I}_n-\mathbf{A}_n(\lambda)])^2}{n\|(\mathbf{I}_n-\mathbf{A}_n(\lambda))\mathbf{y}\|^2} - n^{-1}\|\varepsilon\|^2 + \sigma^2\right|}{r_n(\lambda)} \\
&\leq 2\frac{\sigma^2\mathrm{tr}[\mathbf{I}_n-\mathbf{A}_n(\lambda)]}{\|(\mathbf{I}_n-\mathbf{A}_n(\lambda))\mathbf{y}\|^2}\cdot\frac{n^{-1}\left|\mathbf{f}^T(\mathbf{I}_n-\mathbf{A}_n(\lambda))\varepsilon\right|}{r_n(\lambda)} \\
&\quad +2\frac{\sigma^2\mathrm{tr}[\mathbf{I}_n-\mathbf{A}_n(\lambda)]}{\|(\mathbf{I}_n-\mathbf{A}_n(\lambda))\mathbf{y}\|^2}\cdot\frac{n^{-1}\left|\varepsilon^T\mathbf{A}_n(\lambda)\varepsilon-\sigma^2\mathrm{tr}[\mathbf{A}_n(\lambda)]\right|}{r_n(\lambda)} \\
&\quad +2\frac{\left|(\frac{\sigma^2\mathrm{tr}[\mathbf{I}_n-\mathbf{A}_n(\lambda)]}{\|(\mathbf{I}_n-\mathbf{A}_n(\lambda))\mathbf{y}\|^2}-1)(\sigma^2-n^{-1}\|\varepsilon\|^2)\right|}{r_n(\lambda)}.
\end{aligned} \tag{4.6.14}$$

Note that $n^{-1}\mathrm{tr}[\mathbf{I}_n - \mathbf{A}_n(\lambda_n)] \to 1$, $n^{-1}\|(\mathbf{I}_n - \mathbf{A}_n(\lambda_n))\mathbf{y}\|^2 \to \sigma^2$ from the Lemma 4.3.7, and $\sup_{\lambda>0}\left|\frac{r_n(\lambda)}{E[r_n(\lambda)]} - 1\right| \to 0$ by the Lemma 4.3.5. Thus it suffices for us to show the following three equations

$$\sup_{\lambda>0}\frac{n^{-1}\left|\mathbf{f}^T(\mathbf{I}_n-\mathbf{A}_n(\lambda))\varepsilon\right|}{E[r_n(\lambda)]} \to 0, \tag{4.6.15}$$

$$\sup_{\lambda>0}\frac{n^{-1}\left|\varepsilon^T\mathbf{A}_n(\lambda)\varepsilon-\sigma^2\mathrm{tr}[\mathbf{A}_n(\lambda)]\right|}{E[r_n(\lambda)]} \to 0, \tag{4.6.16}$$

$$\sup_{\lambda>0} \frac{\left|(\sigma^2 n^{-1}\mathrm{tr}[\mathbf{I}_n - \mathbf{A}_n(\lambda)] - n^{-1}\|(\mathbf{I}_n - \mathbf{A}_n(\lambda))\mathbf{y}\|^2)(\sigma^2 - n^{-1}\|\varepsilon\|^2)\right|}{E[r_n(\lambda)]} \to 0. \tag{4.6.17}$$

For (4.6.15), according to the Chebyshev inequality, we have for any given $\delta > 0$

$$\begin{aligned}
&\Pr\{\frac{n^{-1}|\mathbf{f}^T(\mathbf{I}_n - \mathbf{A}_n(\lambda))\varepsilon|}{E[r_n(\lambda)]} > \delta\} \\
&\le \delta^{-2}(nE[r_n(\lambda)])^{-2}E\left[(\mathbf{f}^T(\mathbf{I}_n - \mathbf{A}_n(\lambda))\varepsilon)^2\right] \\
&= \delta^{-2}(nE[r_n(\lambda)])^{-2}\sigma^2\mathrm{tr}\left[(\mathbf{I}_n - \mathbf{A}_n(\lambda))\mathbf{f}\,\mathbf{f}^T(\mathbf{I}_n - \mathbf{A}_n(\lambda))\right] \\
&= \delta^{-2}(nE[r_n(\lambda)])^{-2}\sigma^2\|(\mathbf{I}_n - \mathbf{A}_n(\lambda))\mathbf{f}\|^2 \\
&= \delta^{-2}(nE[r_n(\lambda)])^{-1}\sigma^2\frac{\|(\mathbf{I}_n - \mathbf{A}_n(\lambda))\mathbf{f}\|^2}{nE[r_n(\lambda)]} \\
&\le \delta^{-2}\sigma^2(nE[r_n(\lambda)])^{-1} \to 0,
\end{aligned}$$

since $nE[r_n(\lambda)] \ge \|(\mathbf{I}_n - \mathbf{A}_n(\lambda))\mathbf{f}\|^2$.

For (4.6.16), again using the Chebyshev inequality, we have for any given $\delta > 0$

$$\begin{aligned}
&\Pr\{\frac{n^{-1}\left|\varepsilon^T\mathbf{A}_n(\lambda)\varepsilon - \sigma^2\mathrm{tr}[\mathbf{A}_n(\lambda)]\right|}{E[r_n(\lambda)]} > \delta\} \\
&\le \delta^{-2}(nE[r_n(\lambda)])^{-2}E\Big[(\varepsilon^T\mathbf{A}_n(\lambda)\varepsilon - \sigma^2\mathrm{tr}[\mathbf{A}_n(\lambda)])^2\Big] \\
&= \delta^{-2}(nE[r_n(\lambda)])^{-1}\frac{E\left[(\varepsilon^T\mathbf{A}_n(\lambda)\varepsilon)^2\right] - \left(\sigma^2\mathrm{tr}[\mathbf{A}_n(\lambda)]\right)^2}{nE[r_n(\lambda)]}.
\end{aligned}$$

Since $nE[r_n(\lambda)] \ge \sigma^2\mathrm{tr}[\mathbf{A}_n(\lambda)^2]$, we only need to show

$$\frac{E\left[(\varepsilon^T\mathbf{A}_n(\lambda)\varepsilon)^2\right] - \left(\sigma^2\mathrm{tr}[\mathbf{A}_n(\lambda)]\right)^2}{\sigma^2\mathrm{tr}[\mathbf{A}_n(\lambda)^2]} < \text{Constant}. \tag{4.6.18}$$

Denote $\mathbf{A}_n(\lambda) = (A_{ij})_{n\times n}$, then we have

$$\begin{aligned}
E\left[(\varepsilon^T\mathbf{A}_n(\lambda)\varepsilon)^2\right] &= E[(\sum_{i,j}A_{ij}\varepsilon_i\varepsilon_j)(\sum_{i',j'}A_{i'j'}\varepsilon_{i'}\varepsilon_{j'})] \\
&= E[(\sum_i A_{ii}\varepsilon_i^2)(\sum_{i'}A_{i'i'}\varepsilon_{i'}^2)] + E[(\sum_{i\ne j}A_{ij}\varepsilon_i\varepsilon_j)(\sum_{i'\ne j'}A_{i'j'}\varepsilon_{i'}\varepsilon_{j'})] \\
&\le \left(\sum_{i=1}^n A_{ii}\sigma^2\right)^2 + \sum_{i=1}^n A_{ii}^2E[\varepsilon_i^4] + \sum_{i\ne j}A_{ij}^2\sigma^4.
\end{aligned}$$

There exists a constant c such that $E[\varepsilon_i^4] \le c\sigma^2$ and $\sigma^4 \le c\sigma^2$, so we get

$$\begin{aligned}
E\left[(\varepsilon^T\mathbf{A}_n(\lambda)\varepsilon)^2\right] &\le \left(\sum_{i=1}^n A_{ii}\sigma^2\right)^2 + c\sum_{i=1}^n A_{ii}^2\sigma^2 + c\sum_{i\ne j}A_{ij}^2\sigma^2 \\
&= \left(\sum_{i=1}^n A_{ii}\sigma^2\right)^2 + c\sum_{i,j}A_{ij}^2\sigma^2 \\
&= \left(\sigma^2\mathrm{tr}[\mathbf{A}_n(\lambda)]\right)^2 + c\sigma^2\mathrm{tr}[\mathbf{A}_n(\lambda)^2],
\end{aligned}$$

which implies (4.6.18), and immediately leads to (4.6.16).

For (4.6.17), using the proved (4.6.15), (4.6.16) and $\sigma^2(n^{-1}\mathrm{tr}[\mathbf{A}_n(\lambda)])^2 \le \sigma^2 n^{-1}\mathrm{tr}[\mathbf{A}_n(\lambda)^2] \le E[r_n(\lambda)]$, we only need to show

$$\sup_{\lambda>0} \frac{\left|\sigma^2 - n^{-1}\|\varepsilon\|^2\right|}{(E[r_n(\lambda)])^{1/2}} \to 0, \tag{4.6.19}$$

since the fact that

$$\begin{aligned}
&\left|\sigma^2 n^{-1}\mathrm{tr}[\mathbf{I}_n - \mathbf{A}_n(\lambda)] - n^{-1}\|(\mathbf{I}_n - \mathbf{A}_n(\lambda))\|^2\right| \\
&= \left|\sigma^2 - \sigma^2 n^{-1}\mathrm{tr}[\mathbf{A}_n(\lambda)] - n^{-1}\|\varepsilon + \mathbf{f} - \hat{\mathbf{f}}_n(\lambda)\|^2\right| \\
&= |\sigma^2 - \sigma^2 n^{-1}\mathrm{tr}[\mathbf{A}_n(\lambda)] - n^{-1}\|\varepsilon\|^2 - r_n(\lambda) - 2n^{-1}(\mathbf{f} - \hat{\mathbf{f}}_n(\lambda))^T\varepsilon| \\
&= |\sigma^2 - n^{-1}\|\varepsilon\|^2 - \sigma^2 n^{-1}\mathrm{tr}[\mathbf{A}_n(\lambda)] - r_n(\lambda) - 2n^{-1}\mathbf{f}^T(\mathbf{I}_n - \mathbf{A}_n(\lambda))\varepsilon + 2n^{-1}\varepsilon^T\mathbf{A}_n(\lambda)\varepsilon| \\
&\le \left|\sigma^2 - n^{-1}\|\varepsilon\|^2\right| + r_n(\lambda) + 2n^{-1}\left|\mathbf{f}^T(\mathbf{I}_n - \mathbf{A}_n(\lambda))\varepsilon\right| + 2n^{-1}\left|\varepsilon^T\mathbf{A}_n(\lambda)\varepsilon - \sigma^2\mathrm{tr}[\mathbf{A}_n(\lambda)]\right| \\
&\quad + \sigma^2 n^{-1}\mathrm{tr}[\mathbf{A}_n(\lambda)].
\end{aligned}$$

By the Chebyshev inequality, we have for any given $\delta > 0$

$$\begin{aligned}
&\Pr\{\frac{\left|\sigma^2 - n^{-1}\|\varepsilon\|^2\right|}{(E[r_n(\lambda)])^{1/2}} > \delta\} \\
&\le \delta^{-2}(E[r_n(\lambda)])^{-1}E\left[(\sigma^2 - n^{-1}\|\varepsilon\|^2)^2\right] \\
&= \delta^{-2}(E[r_n(\lambda)])^{-1}\left(n^{-2}E[\|\varepsilon\|^4] - \sigma^4\right) \\
&\le \delta^{-2}(E[r_n(\lambda)])^{-1}\left(n^{-2}(n^2\sigma^4 + nE[\varepsilon_i^4]) - \sigma^4\right) \\
&= \delta^{-2}(nE[r_n(\lambda)])^{-1}E[\varepsilon_i^4] \to 0,
\end{aligned}$$

which implies (4.6.19).

Now it remains to prove (4.3.18), the numerator of which can be rearranged as

$$\begin{aligned}
&n^{-1}\|\tilde{\mathbf{f}}_n(\hat{\lambda}) - \hat{\mathbf{f}}_n(\hat{\lambda})\|^2 \\
&= \left(\frac{\sigma^2 n^{-1}\mathrm{tr}[\mathbf{I}_n - \mathbf{A}_n(\lambda)]}{n^{-1}\|(\mathbf{I}_n - \mathbf{A}_n(\lambda))\mathbf{y}\|^2} - 1\right)^2 n^{-1}\|(\mathbf{I}_n - \mathbf{A}_n(\lambda))\mathbf{y}\|^2 \\
&= \frac{\left((\sigma^2 - n^{-1}\|\varepsilon\|^2) - r_n(\lambda) - 2n^{-1}\mathbf{f}^T(\mathbf{I}_n - \mathbf{A}_n(\lambda))\varepsilon + 2n^{-1}(\varepsilon^T\mathbf{A}_n(\lambda)\varepsilon - \sigma^2\mathrm{tr}[\mathbf{A}_n(\lambda)]) + \sigma^2 n^{-1}\mathrm{tr}[\mathbf{A}_n(\lambda)]\right)^2}{n^{-1}\|(\mathbf{I}_n - \mathbf{A}_n(\lambda))\mathbf{y}\|^2}.
\end{aligned}$$

To get (4.3.18), since $n^{-1}\|(\mathbf{I}_n - \mathbf{A}_n(\lambda))\mathbf{y}\|^2 \to \sigma^2$, it suffices to show the following

$$\frac{\left(\sigma^2 - n^{-1}\|\varepsilon\|^2\right)^2}{r_n(\lambda)} \to 0, \tag{4.6.20}$$

$$\frac{\left(n^{-1}\mathbf{f}^T(\mathbf{I}_n - \mathbf{A}_n(\lambda))\varepsilon\right)^2}{r_n(\lambda)} \to 0, \tag{4.6.21}$$

$$\frac{\left(n^{-1}(\varepsilon^T\mathbf{A}_n(\lambda)\varepsilon - \sigma^2\mathrm{tr}[\mathbf{A}_n(\lambda)])\right)^2}{r_n(\lambda)} \to 0, \tag{4.6.22}$$

$$\frac{\left(n^{-1}\mathrm{tr}[\mathbf{A}_n(\lambda)]\right)^2}{r_n(\lambda)} \to 0. \tag{4.6.23}$$

Note that $\sup_{\lambda>0}\left|\frac{r_n(\lambda)}{E[r_n(\lambda)]} - 1\right| \to 0$, then (4.6.20), (4.6.21) and (4.6.22) can be easily proved from (4.6.19), (4.6.15) and (4.6.16) respectively. The last equation (4.6.23) follows from $\sigma^2 n^{-1}\mathrm{tr}[\mathbf{A}_n(\lambda)^2] \le E[r_n(\lambda)]$ and (4.3.16).

Hence, we complete the proof of Lemma 4.3.9. □

4.6.15 Proof of Lemma 4.3.10

In order to prove Lemma 4.3.10, we need the following lemmas.

Lemma 4.6.1. *Under the condition (A.2), we have $\tilde{r}_n(\lambda_n) \to 0$ when λ_n is from (A.2).*

Proof. To get $\tilde{r}_n(\lambda_n) \to 0$, it suffices for us to show that

$$\frac{\|(\mathbf{I}_n - \mathbf{A}_n(\lambda_n))\mathbf{y}\|^2}{\mathrm{tr}[\mathbf{I}_n - \mathbf{A}_n(\lambda_n)]} \to \sigma^2, \tag{4.6.24}$$

since the following derivation

$$\begin{aligned}
\tilde{r}_n(\lambda_n) &= n^{-1}\|\tilde{\mathbf{f}}_n(\lambda_n) - \mathbf{f}\|^2 \\
&= n^{-1}\|\varepsilon - \sigma^2\tfrac{\mathrm{tr}[\mathbf{I}_n-\mathbf{A}_n(\lambda_n)]}{\|(\mathbf{I}_n-\mathbf{A}_n(\lambda_n))\mathbf{y}\|^2}(\mathbf{I}_n - \mathbf{A}_n(\lambda_n))\mathbf{y}\|^2 \\
&= n^{-1}\|\varepsilon - \sigma^2\tfrac{\mathrm{tr}[\mathbf{I}_n-\mathbf{A}_n(\lambda_n)]}{\|(\mathbf{I}_n-\mathbf{A}_n(\lambda_n))\mathbf{y}\|^2}(\varepsilon + \mathbf{f} - \hat{\mathbf{f}}_n(\lambda_n))\|^2 \\
&\le n^{-1}(1 - \sigma^2\tfrac{\mathrm{tr}[\mathbf{I}_n-\mathbf{A}_n(\lambda_n)]}{\|(\mathbf{I}_n-\mathbf{A}_n(\lambda_n))\mathbf{y}\|^2})^2\|\varepsilon\|^2 \\
&\quad +2n^{-1}\left|1 - \sigma^2\tfrac{\mathrm{tr}[\mathbf{I}_n-\mathbf{A}_n(\lambda_n)]}{\|(\mathbf{I}_n-\mathbf{A}_n(\lambda_n))\mathbf{y}\|^2}\right|\|\varepsilon\|\|\mathbf{f} - \hat{\mathbf{f}}_n(\lambda_n)\| \\
&\quad +n^{-1}(\sigma^2\tfrac{\mathrm{tr}[\mathbf{I}_n-\mathbf{A}_n(\lambda_n)]}{\|(\mathbf{I}_n-\mathbf{A}_n(\lambda_n))\mathbf{y}\|^2})^2\|\mathbf{f} - \hat{\mathbf{f}}_n(\lambda_n)\|^2.
\end{aligned} \tag{4.6.25}$$

Obviously, (4.6.24) follows from the Lemma 4.3.7. □

Lemma 4.6.2. *Under the condition (A.2), we have $\tilde{r}_n(\hat{\lambda}_G) \to 0$.*

Proof. From the uniform consistency of $\text{SURE}_n(\lambda)$ together with the fact that $\hat{\lambda}_G$ minimizes $\text{SURE}_n(\lambda)$, we have

$$\begin{aligned} \tilde{r}_n(\hat{\lambda}_G) &= \text{SURE}_n(\hat{\lambda}_G) + o_p(1) \\ &\leq \text{SURE}_n(\lambda_n) + o_p(1) \\ &= \tilde{r}_n(\lambda_n) + o_p(1) \\ &= o_p(1) \end{aligned} \tag{4.6.26}$$

This is equivalent to say $\tilde{r}_n(\hat{\lambda}_G) \to 0$. □

Lemma 4.6.3. *Under the condition (A.2), we have* $\text{GCV}_n(\hat{\lambda}_G) \to \sigma^2$.

Proof. This is trivial from lemma 4.6.2. □

Lemma 4.6.4. *If ε_i's are i.i.d $N(0, \sigma^2)$, for any $\delta > 0$, we have*

$$\lim_{n\to\infty} \Pr\{\frac{\|(\mathbf{I}_n - \mathbf{A}_n(\hat{\lambda}_G))\mathbf{y}\|^2}{\|(\mathbf{I}_n - \mathbf{A}_n(\hat{\lambda}_G))\mathbf{f}\|^2 + \sigma^2 \text{tr}[(\mathbf{I}_n - \mathbf{A}_n(\hat{\lambda}_G))^2]} \leq 1 - \delta\} = 0. \tag{4.6.27}$$

Proof. According to the proof of Lemma 5.2 in [31], the above lemma can be established directly. □

Lemma 4.6.5. *For any sequence $\{\lambda_n\}$ such that*

$$\text{GCV}_n(\lambda_n) \to \sigma^2, \tag{4.6.28}$$

under the condition (A.3) we have $n^{-1}\text{tr}[\mathbf{A}_n(\lambda_n)] \to 0$.

Proof. Using (4.6.28) and (4.6.27), we have

$$\left(n^{-1}\text{tr}[\mathbf{I}_n - \mathbf{A}_n(\hat{\lambda}_n)]\right)^2 \geq \left(n^{-1}\text{tr}[(\mathbf{I}_n - \mathbf{A}_n(\hat{\lambda}_n))^2]\right)(1 - o_p(1)). \tag{4.6.29}$$

With the fact that $\left(n^{-1}\text{tr}[\mathbf{I}_n - \mathbf{A}_n(\hat{\lambda}_n)]\right)^2 \leq n^{-1}\text{tr}[(\mathbf{I}_n - \mathbf{A}_n(\hat{\lambda}_n))^2]$, then we get

$$\frac{\left(n^{-1}\text{tr}[\mathbf{I}_n - \mathbf{A}_n(\hat{\lambda}_n)]\right)^2}{n^{-1}\text{tr}[(\mathbf{I}_n - \mathbf{A}_n(\hat{\lambda}_n))^2]} \to 1. \tag{4.6.30}$$

Recall $\mathbf{A}_n(\hat{\lambda}) = (\mathbf{I}_n + \hat{\lambda}_n\mathbf{M})^{-1} = (\mathbf{I}_n + \mathbf{K}_n(\hat{\lambda}_n))^{-1}$ and $0 \le \kappa_1 \le \cdots \le \kappa_n$ are the eigenvalues of $\mathbf{K}_n(\hat{\lambda}_n)$. It is clear that $\mathbf{I}_n - \mathbf{A}_n(\hat{\lambda}_n)$ have eigenvalues $\{\frac{\kappa_i}{1+\kappa_i}\}$. Similarly as in [31], let κ be the random variable taking values κ_i with probability n^{-1} for each $i \in \{1, \cdots, n\}$. Then (4.6.30) means $\dfrac{\left(E[\kappa(1+\kappa)^{-1}]\right)^2}{E[\kappa^2(1+\kappa)^{-2}]} \to 1$, and immediately leads to

$$\frac{\kappa(1+\kappa)^{-1}}{E[\kappa(1+\kappa)^{-1}]} \to 1. \tag{4.6.31}$$

Since (4.6.31) implies that both $\kappa_{[pn]}(1+\kappa_{[pn]})^{-1}$ and $\kappa_{[qn]}(1+\kappa_{[qn]})^{-1}$ tend to $E[\kappa(1+\kappa)^{-1}]$, we have $E[\kappa(1+\kappa)^{-1}] \to 1$([22]), from which $n^{-1}\mathrm{tr}[\mathbf{A}_n(\hat{\lambda}_n)] \to 0$ follows. □

Lemma 4.6.6. *For sequence* $\{\lambda_n\}$ *such that* $\mathrm{GCV}_n(\lambda_n) \to \sigma^2$, $\hat{\mathbf{f}}_n(\lambda_n)$ *is consistent iff* $n^{-1}\mathrm{tr}[\mathbf{A}_n(\lambda_n)] \to 0$.

Proof. If $\hat{\mathbf{f}}_n(\lambda_n)$ is consistent, $r_n(\lambda_n) \to 0$ and hence $n^{-1}\|\mathbf{y} - \hat{\mathbf{f}}_n(\lambda_n)\|^2 \to \sigma^2$ since $n^{-1}\|\varepsilon\|^2$. Then from the fact that $\mathrm{GCV}_n(\lambda_n) = \frac{n^{-1}\|\mathbf{y}-\hat{\mathbf{f}}_n(\lambda_n)\|^2}{\left(n^{-1}\mathrm{tr}[\mathbf{I}_n-\mathbf{A}_n(\lambda_n)]\right)^2} \to \sigma^2$, we have $\left(n^{-1}\mathrm{tr}[\mathbf{I}_n - \mathbf{A}_n(\lambda_n)]\right)^2 \to 1$ and thus $n^{-1}\mathrm{tr}[\mathbf{A}_n(\lambda_n)] \to 0$.

Conversely, if $n^{-1}\mathrm{tr}[\mathbf{A}_n(\lambda_n)] \to 0$, since $\mathrm{GCV}_n(\lambda_n) \to \sigma^2$ we have $n^{-1}\|\mathbf{y} - \hat{\mathbf{f}}_n(\lambda_n)\|^2 \to \sigma^2$. Then with the fact that $n^{-1}\|\varepsilon\|^2 \to \sigma^2$, we have $r_n(\lambda_n) \to 0$, which implies $\hat{\mathbf{f}}_n(\lambda_n)$ is consistent. □

From Lemmas 4.6.3, 4.6.5 and 4.6.6, Lemma 4.3.10 is proved.

4.6.16 Proof of Theorem 4.3.6

Proof. From the condition (A.2), for λ_n^* that is the minimizer of $r_n(\lambda)$, we have $r_n(\lambda_n^*) \to 0$. According to Lemma 4.3.8, we have

$$\frac{\left(n^{-1}\mathrm{tr}[\mathbf{A}_n(\lambda_n^*)]\right)^2}{n^{-1}\mathrm{tr}[\mathbf{A}_n(\lambda_n^*)^2]} \to 0. \tag{4.6.32}$$

Hence from Lemma 4.3.9, we have $\mathrm{SURE}_n(\lambda_n^*) - n^{-1}\|\varepsilon_n\|^2 + \sigma^2 = r_n(\lambda_n^*)(1 + o_p(1))$.

On the other hand, from Lemma 4.3.10 this also holds for $\hat{\lambda} = \hat{\lambda}_G$. Therefore we have

$$\mathrm{SURE}_n(\hat{\lambda}_G) - n^{-1}\|\varepsilon_n\|^2 + \sigma^2 = r_n(\hat{\lambda}_G)(1 + o_p(1)) \tag{4.6.33}$$

Since $\mathrm{SURE}_n(\hat{\lambda}_G) \leq \mathrm{SURE}_n(\lambda_n^*)$ and $r_n(\lambda_n^*) \leq r_n(\hat{\lambda}_G)$, we have $r_n(\hat{\lambda}_G)/r_n(\lambda_n^*) \to 1$ in probability. □

CHAPTER V

UNCERTAINTY QUANTIFICATION OF LOCAL WIND SPEED AND WIND PRESSURE COEFFICIENT

5.1 Introduction

There is an increasing need to perform uncertainty analysis (UA) of building performance. Such analysis is, for instance, warranted to support risk conscious decision making in building design and retrofit when decisions are driven by return on investment expectations, or when energy savings guarantees are part of the performance contract. In current practice a building simulation is routinely performed with best guesses of input parameters whose true value cannot be known exactly. Obviously, these guesses directly affect the accuracy and reliability of the outcomes. Although best guesses can possibly lead to simulation outputs whose mean roughly corresponds to the mean of the true outcomes, the true variability of all possible outcomes is not discovered unless a full UA is conducted. Instead of taking best guesses, UA considers input parameters as uncertain and propagates the uncertainty through the model by sampling from the distribution of these input parameters. A general procedure of UA in building performance can be found in [13].

A building simulation tool is a collection of many problem modules that work together to calculate final outcomes. Each module performs a specific function. It is noteworthy that uncertainty existing in any module has many origins: physical parameter uncertainty, module inadequacy (i.e., modeling errors, also referred to as "code errors"), etc. Physical parameter uncertainty reflects the variation of parameters under specified conditions. Even if physical parameter uncertainty is ruled out, i.e., all required input parameters can be assigned the true values, the prediction will not equal the true value of the outcome as there will always be a certain level of model inadequacy.

Microclimate conditions, typically expressed in microclimate variables that enter into the boundary conditions of the building shell, are an important module in simulating energy consumptions. In this chapter, I focus on the physical uncertainty in microclimate conditions, specifically the uncertainty of local wind speed and wind pressure coefficient. In most building simulation tools these parameters are obtained by a preprocessing step which transforms meteorological station weather data to building microclimate parameters. Each simulation tool may deploy its own flavor of preprocessing calculation but the differences across the current tools are not significant. I will refer to the current preprocessing as the "standard model" in this chapter. The standard model implemented in current simulation tools for microclimate conditions is rather crude and its representation of urban surroundings lacks sufficient detail. The UA in this chapter is not limited to specific simulation software, but as "EnergyPlus" is regarded as a representative high-end simulation application, it is used as the starting point for the standard model of microclimate parameters.

The main idea of UA in this chapter is to compare results from the standard model with those from a higher fidelity model, the so-called meso-scale model. By doing this, we can quantify the uncertainty stemming from module inadequacy as well as insufficient knowledge. The major purpose of this UA approach is twofold: (1) to propagate the uncertainty through the simulation model of the whole building and quantify uncertainty in an outcome such as total consumed energy; (2) to enable the ranking of all microclimate parameters on their impact on a certain outcome, accomplished through sensitivity analysis. The latter is an important result as it implicitly shows which parts of the standard model may need improvement to increase the fidelity of the simulation tool. Statistical models are built as connections between the standard model and the meso-scale model, which will make the UQ explicit and facilitate the sensitivity analysis and ultimately the improvement of modules in the simulation tool.

The remainder of the chapter is organized as follows. A brief introduction to building

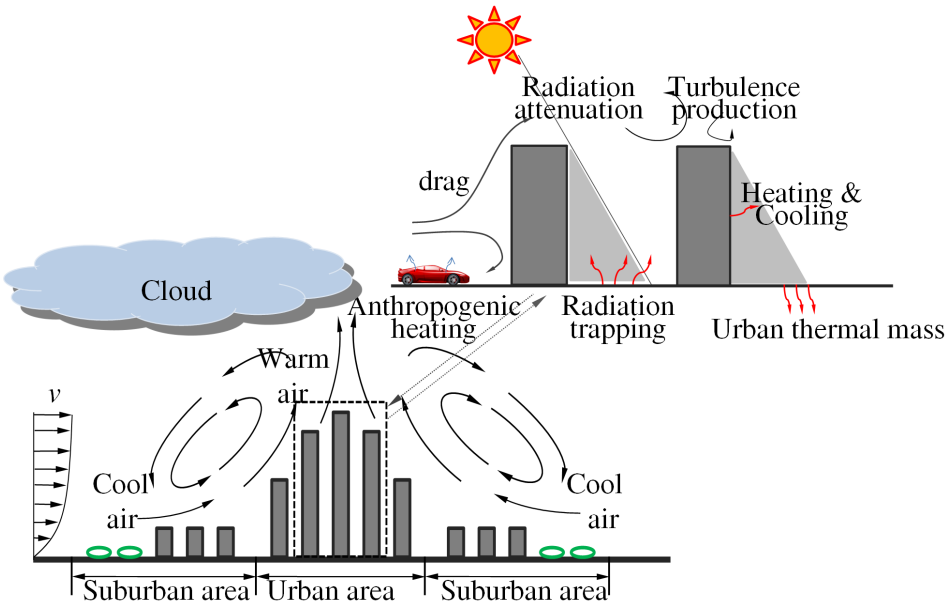

Figure 5.2.1: Suburban and urban climate

microclimate is given in Section 5.2. UA of local wind speed and wind pressure coefficient are then given in Section 5.3. The chapter is concluded with a discussion in Section 5.4.

5.2 Building microclimate

Information about building microclimate is usually unavailable, thus requiring some form of preprocessing calculation to transform recorded weather data from a nearby meteorological station to microclimate conditions. However, as shown in Figure 5.2.1, the microclimate around the building is affected by various factors such as surrounding vegetation, location of neighboring objects, meso-scale air flow patterns, etc. These effects result in significant discrepancy between meteorological station weather data and actual microclimate conditions. In order to derive the building microclimate conditions from nearby meteorological weather data, the preprocessing calculation using the standard model needs to accurately capture air flow patterns around the building, temperature variations due to urban heat island, the existence of complex urban plumes in major metropolitan regions,

and other phenomena. This would require a deep urban representation as well as a computationally intensive model. Although these kinds of models have been well developed at different scales, i.e, for regional weather forecasting ([6]), and urban scale modeling ([53]), they have not been integrated with the current generation of building simulation tools. Quantitative analysis based on statistical models are thus needed to explicitly model the discrepancy between the standard models and the aforementioned higher fidelity models, which would then facilitate the integration of these models into the current generation of simulation tools. In this chapter, I will study two microclimate parameters in detail: (1) local wind speed, (2) wind pressure coefficient. The standard model calculates these variables using weather data from nearby meteorological stations with partial consideration of the urban surrounding of the building under consideration and the distance between meteorological station and the building. It should be noted that the standard model uses some form of urban characterization in the calculation. For instance, the calculation of local wind speed around the building utilizes two coefficients in the transformation of the meteorological weather data to the buildings microclimate wind speed variable.

Unfortunately the standard model for generating the microclimate variables is unreliable and is too crude for investigating the effects of urban surroundings on the local microclimate conditions. So far a rigorous UA of microclimate conditions has not been performed. This is an omission as it is well known that microclimate variables can have significant influence on local temperature, convective heat transfer coefficients, reliance on natural ventilation, and solar heat gains. The motivation for this chapter is to conduct UA for a subset of the microclimate variables, as a preparation towards uncertainty propagation and effect ranking to guide future model improvements.

5.3 UA of local wind speed and wind pressure coefficient

In this section, we show how meso-scale models are used to quantify uncertainty in current standard microclimate models used for the preprocessing in simulation tools. We first

investigate meso-scale models that adequately represent the effects of the urban surroundings on microclimate conditions. Then, we conduct pairwise comparisons of meso-scale model outcomes and standard model outcomes, and analyze their differences. Both the standard model and meso-scale model need meteorological station weather data and an urban built from representation as input. Wind speed and wind pressure coefficient standard models only recognize a crude representation of the urban form specification in the form of a terrain classification defined in the ASHRAE Handbook (i.e., open country, urban and suburban areas and large city center ([3])). This leaves a lot of variability in the urban form of each category. This variability results in uncertainty as the specific built form is not regarded in the standard model. The effect of this variability is examined by generating a set of experimental situations within each terrain classification to explore the effects of all plausible urban contexts. For each experimental situation, we can now calculate the differences between the two model outcomes. Aggregation of all estimates leads to distribution of uncertainty in the two microclimate parameters.

A global data set of urban form and building properties was developed to study the urban climate ([25]). The data set includes the urban morphology and physical characteristics of building materials across 33 regions of the world and subdivides urban areas into four levels of urban density: tall building district, high, medium and low density urban. This study uses this data set to consider the variability of urban form.

5.3.1 UA of local wind speed

Typically simulation software follows Chapter 16 of ASHRAE Fundamentals ([3]) to compute local wind speed in the local terrain. ASHRAE Fundamentals specifies a method that computes local wind speed at a certain height from the measured wind speed V_{met} at the meteorological station with use of a wind reduction factor. The wind reduction factor is derived as a function of measurement height z, wind exponent α, and boundary layer

thickness δ for the site and meteorological station as follows:

$$V_z = V_{met}(\frac{\delta_{met}}{z_{met}})^{\alpha_{met}}(\frac{z}{\delta})^{\alpha}, \tag{5.3.1}$$

where V_z is the local wind speed at measurement height z, V_{met} is the measured wind speed value at the meteorological station, z_{met} is the height at which the measurements are taken at the meteorological station, α_{met} and δ_{met} are the wind profile exponent and boundary layer thickness at the meteorological station respectively, α and δ are the local wind profile exponent and boundary layer thickness respectively. As mentioned before, in the current practice of the standard model, default values are provided for α and δ for three terrain types.

For the meso-scale model, we deploy part of a Community Land Model (CLM) that computes average local wind speed in the urban context ([42]). It approximates urban surroundings, following the urban canopy concept. The CLM urban form parameterization is represented in Figure 5.3.1. All buildings are identical in terms of their geometry, and are regularly distributed over the urban grid. The open space between two rows of buildings is defined as a canyon. This approximation enables 3-D complex urban form representation to be transformed into a 2-D layout. Three geometric variables are used to parameterize urban surroundings: (1) canyon height (H), (2) canyon ratio (H/W), and (3) building length-to-width ratio (B_L/B_W). The urban parameterization scheme is used to capture all variability in heat transfer in the urban context such as drag effects and radiation trapping.

CLM computes average local wind speed in the canyon areas surrounding a building at a given measurement height z. It does not capture variation of local wind speed along building surfaces at a given z. As a result, the difference between the ASHRAE standard model and CLM outcomes only represents uncertainties in average local wind speed (i.e., averaged along the horizontal perimeter of the building shape at a given z).

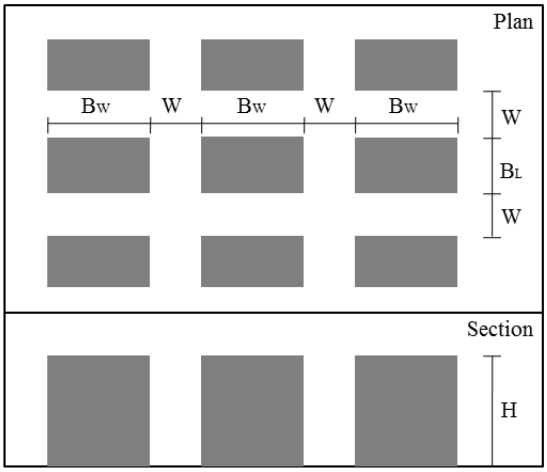

Figure 5.3.1: Urban parameterization scheme in the Community Land Model

5.3.1.1 Design of experiments

Following the terrain classification in the standard model, we generate experimental situations for each terrain separately. Recall that the situations are characterized by canyon height H, canyon ratio H/W and building length-to-width ratio B_L/B_W. For the terrain of urban and city, we use the global data set from [25] for H and H/W and use Latin hypercube sampling ([37]) to generate samples for B_L/B_W. The generated samples of B_L/B_W are then crossed with the real data of H and H/W to form a cross array. For the terrain of open country, no real data is available and Latin hypercube sampling is used to generate samples for the three parameters. A situation is defined by a specific set of values of the three parameters. For each situation, 20 cases are generated for the nearby meteorological station surroundings using Latin hypercube sampling.

5.3.1.2 Statistical analysis

In this section, I build statistical models to analyze the difference between outcomes from the standard model and CLM. I focus on the terrain of city and omit the details for the other two terrains. For mathematical convenience, I model the difference on the log scale. The modeling is done separately for each situation. For each situation, I have 20 observations corresponding to the 20 generated meteorological surroundings. I then summarize the data by its sample mean and sample variance. The purpose is then to model the sample mean and sample variance as a function of the measurement height z. Interestingly, the sample variance is a constant and does not change with respect to z. Therefore, I focus on the modeling of the sample mean.

Through exploratory analysis, the following statistical model is proposed to model the difference as a function of z.

$$Diff = \begin{cases} \beta_{01} + \beta_{11} z + \varepsilon, & z \leq \tau, \\ \beta_{02} + \beta_{12} z + \varepsilon, & z > \tau, \end{cases} \tag{5.3.2}$$

where $\varepsilon \sim N(0, \sigma^2)$. Note that the random error comes from the sampling of meteorological

station surroundings. The variance of the random error is assumed constant since we have the same number of samples for each situation.

Maximum likelihood estimation is used to estimate the unknown parameters in the model. To ensure the continuity of the estimated function, I put the constraint $\beta_{01} + \beta_{11}\tau = \beta_{02} + \beta_{12}\tau$ on the parameters. For the optimization, the difficult part is to estimate τ. I use the brutal force idea here, meaning that I change τ from the smallest observed z value to the largest observed z value. For a given τ, let $\mathbf{Diff}_1 = (Diff_{11}, \ldots, Diff_{1n_1})$ denote the response values whose z values are smaller than or equal to τ. Denote $\mathbf{z}_1 = (z_{11}, \ldots, z_{1n_1})$ as the corresponding vector of measurement heights. Let $\mathbf{Diff}_2 = (Diff_{21}, \ldots, Diff_{2n_2})$ denote the response values whose z values are larger than τ. Denote $\mathbf{z}_2 = (z_{21}, \ldots, z_{2n_2})$ as the corresponding vector of measurement heights. The likelihood function can then be written as follows:

$$\prod_{i=1}^{n_1} \frac{1}{\sqrt{2\pi}\sigma} exp(-\frac{(Diff_{1i} - \beta_{01} - \beta_{11}z_{1i})^2}{2\sigma^2}) \prod_{j=1}^{n_2} \frac{1}{\sqrt{2\pi}\sigma} exp(-\frac{(Diff_{2j} - \beta_{02} - \beta_{12}z_{2j})^2}{2\sigma^2}). \tag{5.3.3}$$

The estimation can then be conducted as in the normal situation. For each choice of τ, I calculate the likelihood function value and choose the one that has the highest likelihood function value.

Since the estimation is done separately for each situation, I get a histogram of each of the parameters for the terrain of city. They are displayed in Figure 5.3.2.

Since we have generated a large amount of data from deterministic simulation, I use R^2 to evaluate the goodness-of-fit of the model in (5.3.2). The R^2 for all the situations is above 0.95. Most situations have an R^2 above 0.99. This indicates that the model in (5.3.2) fits the data very well. Given the fitted model, for any value of measurement height z, a distribution of the difference between the outcomes from the standard model and CLM can be derived. Note that the variation in the distribution represents the variation among different situations as well as the variation among the surroundings of the meteorological station in each situation. By adding the distribution of the difference for each measurement height to

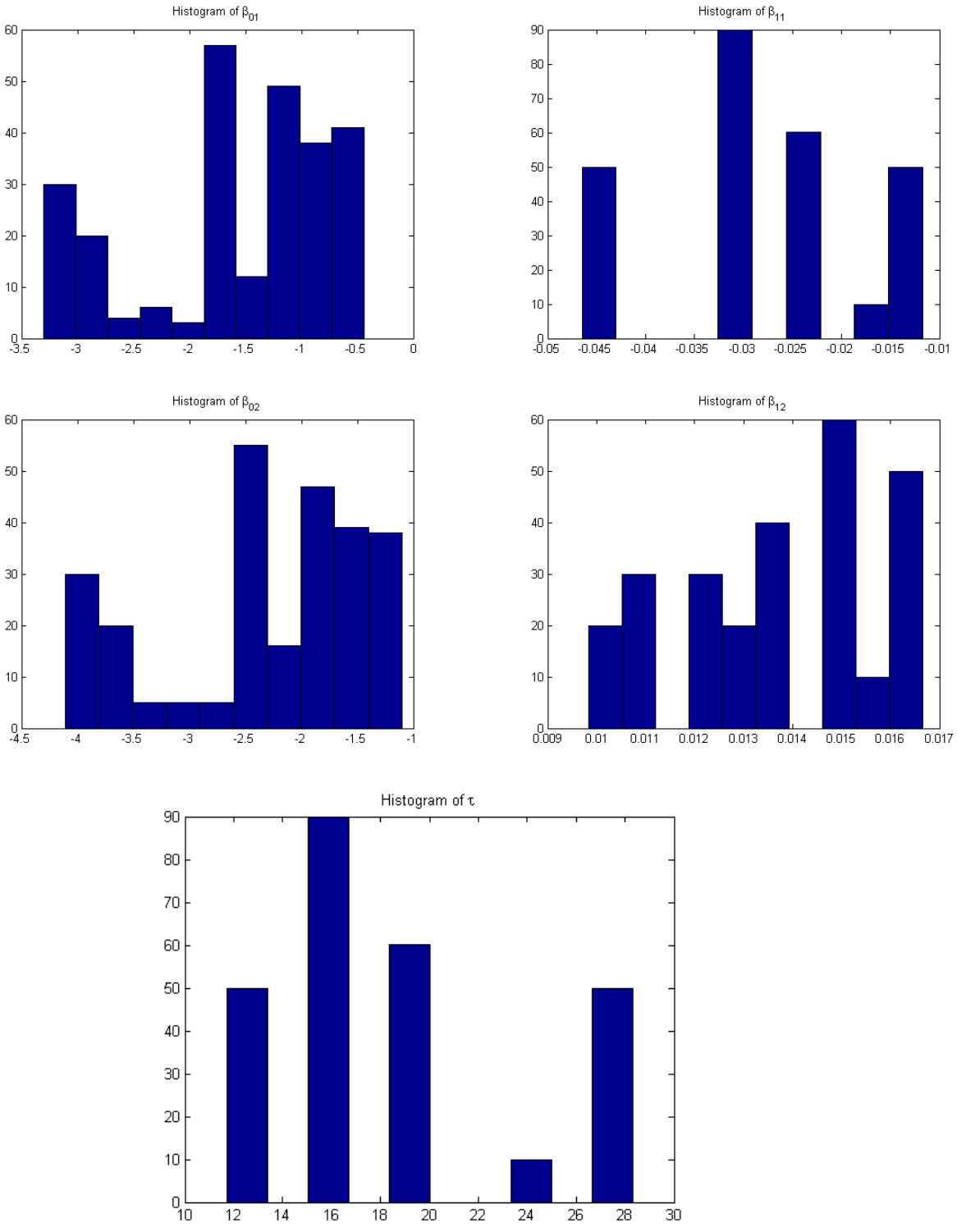

Figure 5.3.2: Histograms of unknown parameters

the outcome of the standard model, the uncertainty in local wind speed is quantified.

5.3.2 UA of wind pressure coefficient

The standard model in most simulation software follows Chapter 16 of ASHRAE Fundamentals ([3]) to compute wind pressure on a building. Wind pressure on a building surface depends on ambient air density ρ, local wind speed V_z and wind surface pressure coefficient C_p as defined in $P_w = C_p \rho \frac{V_z^2}{2}$. In this section, I focus on the UA of C_p. ASHRAE Fundamentals describes empirically driven models that generate surface-averaged wind pressure coefficients for low-rise buildings ([62]) and high-rise buildings ([2]). All models are defined as functions of the wind incident angle θ and the ratio of the width of the wall under consideration to that of the adjacent wall B_W/B_L. The model in [62] is shown in (5.3.4).

$$C_p = 0.6ln(1.248 - 0.703sin(\frac{\theta}{2}) - 1.175sin^2\theta + 0.031sin^2(2\alpha G) \\ +0.769cos(\frac{\theta}{2}) + 0.07G^2sin^2(\frac{\theta}{2}) + 0.717cos^2(\frac{\theta}{2})) \tag{5.3.4}$$

This is the standard model the outcome from which will be compared to a higher fidelity model.

For the higher fidelity model, we use the TNO C_p generator that calculates wind surface pressure coefficients in the urban context. The C_p generator is a parametric model developed based on experimental data that calculates coefficient values based on wind direction and building dimensions and corrects coefficient values, taking the surrounding terrain into account ([30]). The C_p generator requires information about full geometries and terrain roughness for adjacent obstacles and for distant obstacles respectively. As outcomes, we obtain surface-averaged coefficient values for each wall and each wind direction. We follow the urban parameterization scheme in the Climate Land Model to represent surroundings in which a building under consideration may be situated. This leads to the selection of eight identical buildings surrounding the building under consideration. Configuration of the buildings and their spatial relationships are parameterized by four variables: canyon

height H, canyon ratio H/W, building height-to-length ratio H/B_L, and building width-to-length ratio B_W/B_L. Depending on the value of H/B_L, buildings can be classified into high-rise building and low-rise building. In this section, I focus on the low-rise buildings.

5.3.2.1 Design of experiments

Due to the cumbersome preparation for the simulation, we choose only three values for B_W/B_L. That is $B_W/B_L = 1, 2, 4$. For each value of B_W/B_L, a Latin hypercube sample of 27 runs was generated for H, H/W, H/B_L and Roughness. Note that we have real data for H and H/W. In generating samples based on the real data, I used the maximin distance criterion ([27]) to ensure that the design points are spread out over the design region.

5.3.2.2 Statistical analysis

Since we only have three values of B_W/B_L, the statistical modeling is done separately for each value of B_W/B_L. The response is again the difference between the outcomes from the standard model and the C_p calculator. The following statistical model is used.

$$Diff = \begin{cases} f_1(\theta) + \varepsilon_1, & B_W/B_L = 1, \\ f_2(\theta) + \varepsilon_2, & B_W/B_L = 2, \\ f_3(\theta) + \varepsilon_3, & B_W/B_L = 4, \end{cases} \tag{5.3.5}$$

where ε_1 has zero mean and variance σ_1^2, ε_2 has zero mean and variance σ_2^2 and ε_3 has zero mean and variance σ_3^2. To estimate the f_i's, I have tried different nonparametric models such as smoothing spline models ([64]), local polynomial regression models ([7]) and stochastic kriging models ([10]). The results all look similar. The main message is that a large portion of the variation of the pressure coefficient cannot be explained by the incident angle. In order to quantify the effect of B_W/B_L, I propose to fit a global parametric model for the f_i's. By trial and error, I found a fourth-order polynomial regression model is adequate for all the three cases in the sense that the residual behavior is similar to the aforementioned

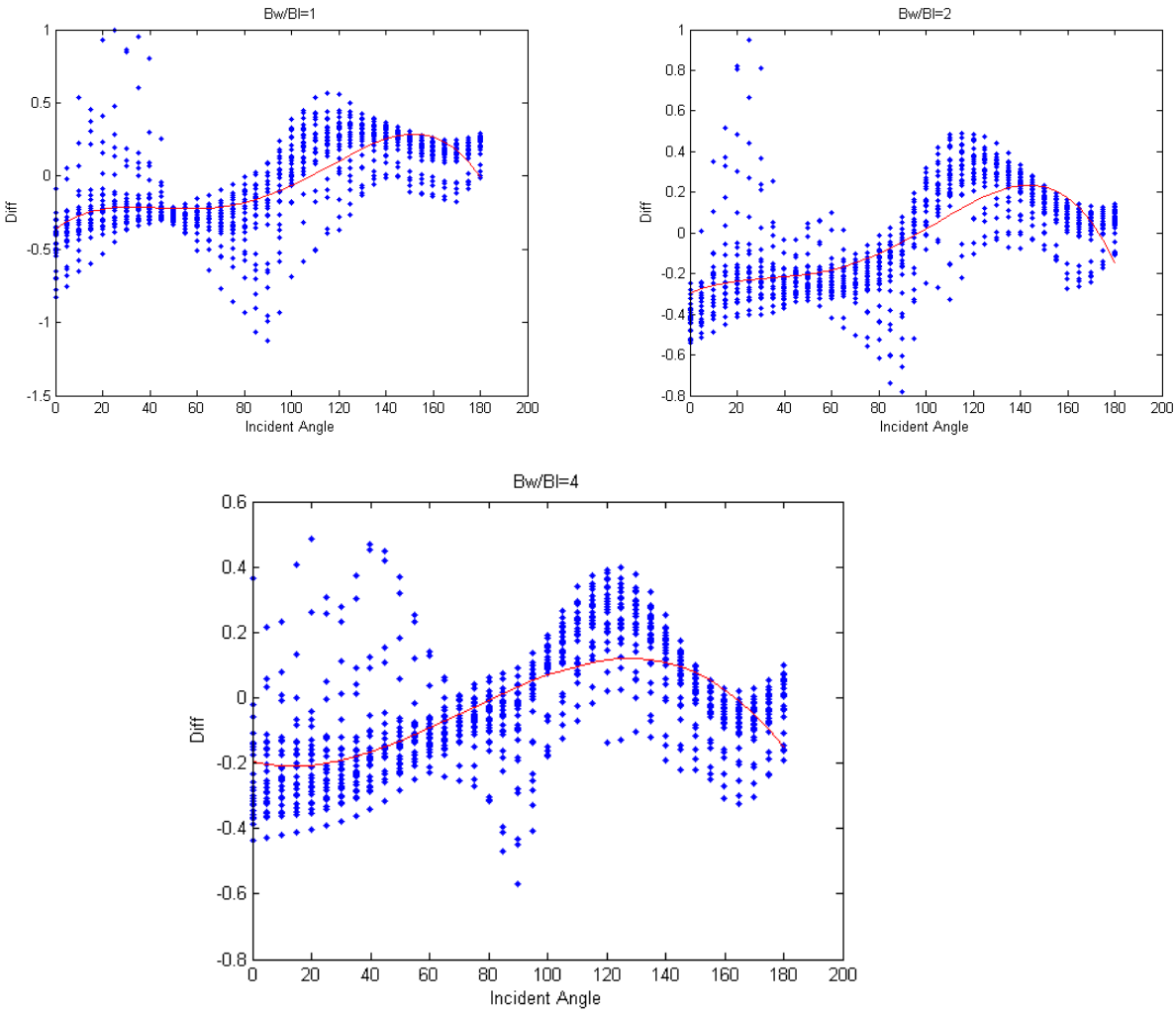

Figure 5.3.3: Fitted polynomial regression models for wind pressure coefficient

nonparametric models. The polynomial regression model is specified as follows:

$$f_i(\theta) = \beta_{0i} + \beta_{1i}\theta + \beta_{2i}\theta^2 + \beta_{3i}\theta^3 + \beta_{4i}\theta^4, \quad i = 1, 2, 3. \tag{5.3.6}$$

The scatter plots of the difference against incident angle as well as the fitted model are shown in Figure 5.3.3. Statistical hypothesis testing confirms that B_W/B_L has a significant effect on the coefficients βs as well as the variances. I propose to use a linear interpolator to obtain the βs and variance for those B_W/B_L values not in the data. As a result, for any value of B_W/B_L, I can obtain a fourth-order polynomial regression model describing the relationship between difference and incident angle. Based on the model, the distribution of

differences between the outcomes from the standard model and the C_p generator for any value of B_W/B_L and incident angle can be derived. Note that the variation of the distribution is induced by the different urban situations in which a low-rise building may be located.

5.4 Discussion

In this chapter, I have quantified the uncertainty of two microclimate parameters: local wind speed and wind pressure coefficient. The main idea is to compare the outcomes from the standard model against those from a higher fidelity model. Statistical models are built to connect the standard model and higher fidelity models, which can then facilitate the improvement of the corresponding modules in current simulation tools. Future research includes propagating the uncertainties in these parameters to a final outcome such as energy consumption and sensitivity analysis of microclimate parameters on their impact on final outcomes such as energy consumption.

REFERENCES

[1] Agmon, S., *Lectures on Elliptic Boundary Value Problems*. D. Van Norstrand, Princeton, New Jersey, 1965.

[2] Akins, R. E., Peterka, J. A., and Cermak, J. E., "Averaged pressure coefficients for rectangular buildings," *Wind Engineering: Proceedings of the Fifth International Conference*, vol. 7, pp. 369–380, 1979.

[3] ASHRAE, *ASHRAE Handbook Fundamentals*. Atlanta, GA: American Society of Heating, Refrigerating and Air-conditioning Engineers, 2005.

[4] Bernardo, M. C., Buck, R., Liu, L., Nazaret, W. A., Sacks, J., and Welch, W. J., "Integrated design optimization using a sequential strategy," *IEEE Transaction on Computer Aided Design*, vol. 11, pp. 361–372, 1992.

[5] Chan, W. M. and George, A., "A linear time implementation of the reverse cuthill-mckee algorithm," *BIT Numerical Mathematics*, vol. 20, no. 1, pp. 8–14, 1980.

[6] Chen, F. e. a., "The integrated wrf/urban modeling system: Development, evaluation, and applications to urban environmental problems," *International Journal of Climatology*, vol. 34, no. 2, pp. 273–288, 2011.

[7] Cleveland, W. S., "Lowess: A program for smoothing scatterplots by robust locally weighted regression," *The American Statistician*, vol. 35, p. 54, 1981.

[8] Cowling, A., Chambers, R., Lindsay, R., and Parameswaran, B., "Applications of spatial smoothing to survey data," *Survey Methodology*, vol. 22, pp. 175–183, 1996.

[9] Cox, D. D., Park, J. S., and Singer, C. E., "A statistical method for tuning a computer code to a data base," *Computational Statistics and Data Analysis*, vol. 37, pp. 77–92, 2001.

[10] Cressie, N., *Statistics for Spatial Data*. NY, USA: Wiley, 1993.

[11] Currin, C., Mitchell, T., Morris, M., and Ylvisaker, D., "Bayesian prediction of deterministic functions, with applications to design and analysis of computer experiments," *Journal of American Statistical Association*, vol. 86, pp. 953–963, 1991.

[12] Cuthill, E. and Mckee, J., "Reducing the bandwidth of sparse symmetric matrices," in *The 24th ACM National Conference*, (NY, USA), pp. 157–172, ACM, 1969.

[13] de Wit, S. and Augenbroe, G., "Analysis of uncertainty in building design evaluations and its implications," *Energy and Buildings*, vol. 34, no. 9, pp. 951–958, 2002.

[14] Deng, X., Lin, C., and Qian, P. Z. G., "Designs for the lasso," tech. rep., 2012.

[15] Deng, X. and Qian, P. Z. G., "Sliced cross-valiation for efficient estimation of the error rate of a classification rule," tech. rep., 2012.

[16] Drew, S. and Homem-de Mello, T., "Some large deviation results for latin hypercube sampling," *Proceedings of the 2005 Winter Simulation Conference*, pp. 674–681, 2005.

[17] Duchon, J., "Splines minimizing rotation invariant seminorms in Sobolev spaces," in *Constructive Theory of Functions of Several Variables*, (Berlin), pp. 85–100, Springer-Verlag, 1977.

[18] Dyn, N. and Levin, D., "Construction of surface spline interpolants of scattered data over finite domains," *Rev. Fr. Autom. Inform. Rech. Oper. Anal. Numer.*, vol. 16, pp. 201–209, 1982.

[19] Fang, K. T., Li, R. Z., and Sudjianto, A., *Design and Modelling for Computer Experiments*. New York: Chapman Hall/CRC Press, 2005.

[20] Fang, K. T., Lin, D. K., Winker, P., and Zhang, Y., "Uniform design: Theory and application," *Technometrics*, vol. 42, pp. 237–248, 2000.

[21] George, A. and Liu, J., *Computer Solution of Large Sparse Positive Definite Systems*. Prentice-Hall, 1981.

[22] Girard, D. A., "Asymptotic optimality of the fast randomized versions of GCV and c_l in ridge regression and regularization," *The Annals of Statistics*, vol. 19, no. 4, pp. 1950–1963, 1991.

[23] Golub, G. H. and van Loan, C. F., *Matrix Computations*. Johns Hopkins University Press, 3rd ed., 1996.

[24] Horgan, G. W., "Using wavelets for data smoothing: A simulation study," *Journal of Applied Statistics*, vol. 26, pp. 923–932, 1999.

[25] Jackson, T. L. e. a., "Parameterization of urban characteristics for global climate modeling," *Annals of the Association of American Geographers*, vol. 100, no. 4, pp. 848–865, 2010.

[26] Jin, R., Chen, W., and Sudjianto, A., "An efficient algorithm for constructing optimal design of computer experiments," *Journal of Statistical Planning and Inference*, vol. 134, pp. 268–287, 2005.

[27] Johnson, M. E., Moore, L. M., and Ylvisaker, D., "Minimax and maximin distance designs," *Journal of Statistical Planning and Inference*, vol. 26, pp. 131–148, 1990.

[28] Jones, D. R., Schonlau, M., and Welch, W. J., "Efficient global optimization of expensive black-box functions," *Journal of Global Optimization*, vol. 13, pp. 455–492, 1998.

[29] Kennedy, M. C. and O'Hagan, A., "Bayesian calibration of computer models (with discussion)," *Journal of Royal Statistical Society, Series B*, vol. 63, pp. 425–464, 2001.

[30] Knoll, B., Phaff, J. C., and de Gids, W. F., "Pressure simulation program," *Proceedings of the Conference on Implementing the Results of Ventilation Research, AIVC*, 1995.

[31] Li, K. C., "From Stein's unbiased risk estimates to the method of generalized cross validation," *The Annals of Statistics*, vol. 13, no. 4, pp. 1352–1377, 1985.

[32] Li, W. and Wu, C. F. J., "Columnwise-pairwise algorithms with applications to the construction of supersaturated designs," *Technometrics*, vol. 39, pp. 171–179, 1997.

[33] Liu, W. H. and Sherman, A. H., "Comparative analysis of the cuthill–mckee and the reverse cuthill–mckee ordering algorithms for sparse matrices," *SIAM Journal on Numerical Analysis*, vol. 13, no. 2, pp. 198–213, 1976.

[34] Loeppky, J. L., Moore, L. M., and Williams, B. J., "Batch sequential designs for computer expriments," *Journal of Statistical Planning and Inference*, vol. 140, pp. 1452–1464, 2010.

[35] Loh, W. L., "On latin hypercube sampling," *Annals of Statistics*, vol. 24, pp. 2058–2080, 1996.

[36] MacKay, D., "Information-based objective functions for active data selection," *Neural Computation*, vol. 4, pp. 589–603, 1992.

[37] McKay, M. D., Beckman, R. J., and Conover, W. J., "A comparison of three methods for selecting values of input variables in the analysis of output from a computer code," *Technometrics*, vol. 21, pp. 239–245, 1979.

[38] Mitchell, T. J. and Morris, M. D., "Bayesian design and analysis of computer experiments: Two examples," *Statistica Sinica*, vol. 2, pp. 359–379, 1992.

[39] Morris, M. D., Mitchell, T. J., and Ylvisaker, D., "Bayesian design and analysis of computer experiments: Use of derivatives in surface prediction," *Technometrics*, vol. 35, pp. 243–255, 1993.

[40] Nussbaum, M., "Spline smoothing in regression models and asymptotic efficiency in L_2," *The Annals of Statistics*, vol. 13, no. 3, pp. 984–997, 1985.

[41] O'Connell, M. A. and Wolfinger, R. D., "Spatial regression models, response surfaces, and process optimization," *Journal of Computational and Graphical Statistics*, vol. 6, pp. 224–241, 1997.

[42] Oleson, K. W. e. a., "An urban parameterization for a global climate model. part ii: Sensitivity to input parameters and the simulated urban head island in offline simulations," *Journal of Applied Meteorology and Climatology*, vol. 47, no. 4, pp. 1061–1076, 2008.

[43] Owen, A. B., "A central limit theorem for latin hypercube sampling," *Journal of Royal Statistical Society, Series B*, vol. 54, pp. 541–551, 1992.

[44] Park, J. S., "Optimal latin-hypercube designs for computer experiments," *Journal of Statistical Planning and Inference*, vol. 39, pp. 95–111, 1994.

[45] Pinsker, M. S., "Optimal filtering of square integrable signals in Gaussian white noise (in Ruassian)," *Problems Inform. Transmission*, vol. 16, no. 2, pp. 52–68, 1980.

[46] Qian, P. Z. G., "Nested latin hypercube designs," *Biometrika*, vol. 96, pp. 957–970, 2009.

[47] Qian, P. Z. G., "Sliced latin hypercube designs," *Journal of American Statistical Association*, 2012. to appear.

[48] Qian, P. Z. G., Ai, M., and Wu, C. F. J., "Construction of nested space-filling designs," *Annals of Statistics*, vol. 37, pp. 3616–3643, 2009.

[49] Qian, P. Z. G. and Wu, C. F. J., "Sliced space-filling designs," *Biometrika*, vol. 96, pp. 945–956, 2009.

[50] Qian, P. Z. G., Wu, H., and Wu, C. F. J., "Gaussian process models for computer experiments with qualitative and quantitative factors," *Technometrics*, vol. 50, pp. 192–204, 2008.

[51] Qian, Z., Seepersad, C. C., Joseph, V. R., Allen, J. K., and Wu, C. F. J., "Building surrogate models based on detailed and approximate simulations," *ASME Transactions Journal of Mechanical Design*, vol. 128, pp. 668–677, 2006.

[52] Ramsay, T., "Spline smoothing over difficult regions," *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, vol. 64, no. 2, pp. 307–319, 2002.

[53] Robinson, D., *Computer Modeling for Sustainable Urban Design: Physical Principles, Methods and Applications*. Routledge, 2011.

[54] Sacks, J. and Schiller, S. B., "Spatial designs," *Statistical Decision Theory and Related Topics IV (Vol. 2)*, pp. 385–399, 1988.

[55] Sacks, J., Schiller, S. B., and Welch, W. J., "Designs for computer experiments," *Technometrics*, vol. 31, pp. 41–47, 1992.

[56] Sacks, J., Welch, W. J., Mitchell, T. J., and Wynn, H. P., "Design and analysis of computer experiments (with discussion)," *Statistical Science*, vol. 4, pp. 409–435, 1989.

[57] Samadiani, E., Joshi, Y., Allen, J. K., and Mistree, F., "Adaptable robust design of multiscale convective systems applied to energy efficient data centers," *Numerical Heat Transfer, Part A: Applications*, vol. 57, pp. 69–100, 2010.

[58] Santner, T. J., Williams, B. J., and Notz, W. I., *The Design and Analysis of Computer Expriments*. New York: Springer, 2003.

[59] Speckman, P., "Spline smoothing and optimal rates of convergence in nonparametric regression models," *The Annals of Statistics*, vol. 13, no. 3, pp. 970–983, 1985.

[60] Stein, M., "Large sample properties of simulations using latin hypercube sampling," *Technometrics*, vol. 29, pp. 143–151, 1987.

[61] Stone, C. J., "Optimal global rates of convergence for nonparametric regression," *The Annals of Statistics*, vol. 10, no. 4, pp. 1040–1053, 1982.

[62] Swami, M. V. and Chandra, S., "Correlations for pressure distribution on buildings and calculation of natural-ventilation airflow," *ASHARE Transactions*, vol. 94, pp. 243–266, 1988.

[63] Utreras, F. I., "Convergence rates for multivariate smoothing spline functions," *Journal of Approximation Theory*, vol. 52, no. 1, pp. 1–27, 1988.

[64] Wahba, G., *Spline Models for Observational Data (CBMS-NSF Regional Conference Series in Applied Mathematics)*. Philadelphia, PA: SIAM: Society for Industrial and Applied Mathematics, 1990.

[65] Wahba, G. and Wendelberger, J., "Some new mathematical methods for variational objective analysis using splines and cross-validation," *Monthly Weather Review*, vol. 108, pp. 1122–1145, 1980.

[66] Wand, M. P. and Jones, M. C., *Kernel Smoothing*. London: Chapman and Hall, 1995.

[67] Whitney, H., "Functions differentiable on the boundaries of regions," *Ann. of Math.*, vol. 35, pp. 482–485, 1934.

[68] Williams, B., Morris, M., and Santner, T., "Using multiple computer models/multiple data sources simultaneously to infer calibration parameters," (San Diego, CA), INFORMS Annual Conference, 2009.

[69] Wood, S. N., "Thin plate regression splines," *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, vol. 65, no. 1, pp. 95–114, 2003.

[70] Wood, S. N., "Soap 0.1-5." http://www.maths.bath.ac.uk/~sw283/simon/software.html, 2009.

[71] Wood, S. N., "R package: mgcv." http://cran.r-project.org/web/packages/mgcv/index.html, 2012. Version 1.7-13.

[72] Wood, S. N., Bravington, M. V., and Hedley, S. L., "Soap film smoothing," *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, vol. 70, no. 5, pp. 931–955, 2008.

[73] Wu, C. F. J. and Ding, Y., "Construction of response surface designs for qualitative and quantitative factors," *Journal of Statistical Planning and Inference*, vol. 71, pp. 331–348, 1998.

[74] Xiong, S., Qian, P. Z. G., and Wu, C. F. J., "Sequential design and analysis of high-accuracy and low-accuracy computer codes," tech. rep., 2012.

[75] Ye, K. Q., Li, W., and Sudjianto, A., "Algorithmic construction of optimal symmetric latin hypercube designs," *Journal of Statistical Planning and Inference*, vol. 90, pp. 145–159, 2000.

[76] Zuppa, C., "Error estimates for moving least-square approximations," *Bull Braz. Math. Soc. (N.S.)*, vol. 34, pp. 231–249, 2003.

Some contributions to Latin hypercube designs, irregular region smoothing and uncertainty quantification

Huizhi Xie

139 Pages

Directed by Professor C. F. Jeff Wu

In the first part of the thesis, we propose a new class of designs called multi-layer sliced Latin hypercube design (DSLHD) for running computer experiments. A general recursive strategy for constructing MLSLHD has been developed. Ordinary Latin hypercube designs and sliced Latin hypercube designs are special cases of MLSLHD with zero and one layer respectively. A special case of MLSLHD with two layers, doubly sliced Latin hypercube design, is studied in detail. The doubly sliced structure of DSLHD allows more flexible batch size than SLHD for collective evaluation of different computer models or batch sequential evaluation of a single computer model. Both finite-sample and asymptotical sampling properties of DSLHD are examined. Numerical experiments are provided to show the advantage of DSLHD over SLHD for both sequential evaluating a single computer model and collective evaluation of different computer models. Other applications of DSLHD include design for Gaussian process modeling with quantitative and qualitative factors, cross-validation, etc. Moreover, we also show the sliced structure, possibly combining with other criteria such as distance-based criteria, can be utilized to sequentially sample from a large spatial data set when we cannot include all the data points for modeling. A data center example is presented to illustrate the idea. The enhanced stochastic evolutionary algorithm is deployed to search for optimal design.

In the second part of the thesis, we propose a new smoothing technique called completely-data-driven smoothing, intended for smoothing over irregular regions. The idea is to replace the penalty term in the smoothing splines by its estimate based on local least squares technique. A close form solution for our approach is derived. The implementation is very easy and computationally efficient. With some regularity assumptions on the input region and

analytical assumptions on the true function, it can be shown that our estimator achieves the optimal convergence rate in general nonparametric regression. The algorithmic parameter that governs the trade-off between the fidelity to the data and the smoothness of the estimated function is chosen by generalized cross validation (GCV). The asymptotic optimality of GCV for choosing the algorithm parameter in our estimator is proved. Numerical experiments show that our method works well for both regular and irregular region smoothing.

The third part of the thesis deals with uncertainty quantification in building energy assessment. In current practice, building simulation is routinely performed with best guesses of input parameters whose true value cannot be known exactly. These guesses affect the accuracy and reliability of the outcomes. There is an increasing need to perform uncertain analysis of those input parameters that are known to have a significant impact on the final outcome. In this part of the thesis, we focus on uncertainty quantification of two microclimate parameters: the local wind speed and the wind pressure coefficient. The idea is to compare the outcome of the standard model with that of a higher fidelity model. Statistical analysis is then conducted to build a connection between these two. The explicit form of statistical models can facilitate the improvement of the corresponding modules in the standard model.